TURING 图灵新知

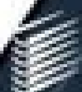
全彩印刷

迷茫的旅行商

一个无处不在的计算机算法问题

[美] William J. Cook 著　　隋春宁 译

In Pursuit of the Traveling Salesman
Mathematics at the Limits of Computation



人民邮电出版社
POSTS & TELECOM PRESS

版权信息

书名：迷茫的旅行商：一个无处不在的计算机算法问题
作者：William J.Cook
译者：隋春宁
ISBN：978-7-115-32773-4

目录

版权声明

序

> “听着，老兄，这片土地上的每一条道路都有我的足迹。”
>
> ——Geoff Mack，《我的足迹遍布四方》

在Geoff Mack的歌曲《我的足迹遍布四方》（*I've Been Everywhere*）中，主人公曾经去过以下城市：里诺、芝加哥、法戈、布法罗、多伦多、温斯洛、萨拉索塔、威奇托、塔尔萨、渥太华、俄克拉荷马、坦帕、巴拿马、马特瓦、拉帕洛马、班戈、巴尔的摩、萨尔瓦多、阿马里洛、托卡皮罗、巴兰基亚、帕迪亚。

1988年2月的一天晚上，我和朋友Vašek Chvátal下定决心追随数学大师的脚步，试着寻找周游多个目的地的最短路线。次日，我们约在曼哈顿下城的一家计算机销售商店碰头，那家店名为Tri-State Camera。店里的技术人员一听说我们是数学家，还想买一台速度快的计算机，便盯着我们的眼睛，告诫道：“你们俩不是想解决那道旅行商问题吧？是不是啊？”他颇有先见之明。在之后的20年里，我们把大部分时光都用在了求解这道题上。期间，有很多台计算机在我们手下逐渐停止运转，就像最初的这一台一样。

这道“臭名昭著”的题是这样的：给定一系列城市，试求出经过所有城市并回到出发地的最短路线。以Geoff Mack歌里唱到的那位旅行者为例，经过那22座城市的路线总共有51 090 942 171 709 440 000条。可以想到，逐一检验所有路线就是一种解题思路。在这样的计算量面前，速度最快的超级计算机也要“挽起袖子”准备辛苦干上一整天。若有足够的耐心，便有希望找到最短路线。然而，假如城市数目变成100座，那么就算征用地球上所有的计算机，也不可能通过逐一检验所有路线找出最短的一条。

最后这句评论针对的是枚举路线的做法，但如果要说旅行商问题确实难解，这理由绝对不足以让人信服。有一些类似的问题其实很容易解决，而它们可供枚举的情形比旅行商的路线还要多。旅行商问题的与众不同之处在于，尽管世界各地的一流应用数学家已经研究了几十年，但人们如今依然不知道，对于一般性的旅行商问题，如何才能得出远远优于简单暴力枚举检验的通用解法。很有可能根本就没有一种高效解法能保证解决该问题的所有具体题目。高效解法是否存在？这是一道严肃的数学问题，涉及可行计算的界限，因此触及复杂性理论的核心。对于想努力求出通用解法的勇士，克雷数学研究所准备了一份大奖——任何人

只要能够提出高效解法或者证明不存在高效解法，便会得到100万美元的奖金。

在旅行商问题研究领域，克雷大奖关注的复杂性问题就像圣杯一样。我们也许连解法的影子都看不到，但这并不意味着数学家至今的研究都一无所获。事实上，这道问题带来了许多美妙而深刻的结论和猜想。在精确计算领域，一道包含85 900座城市的难题于2006年宣告破解。该题的最优路线经过计算脱颖而出，若是对数目异常庞大的所有路线逐一检验，需要用顶尖计算机工作站连续运转136年。在实际应用领域，人们使用该问题的解法，每天可以对大量应用题目计算出最优路线或近优路线。

旅行商问题之所以具有永恒的力量，原因之一在于，它是应用数学领域以及运筹学与数学规划方向的驱动力，对新发现起到了有目共睹的带动作用。而且，更多的发现可能就在前方不远处。本书的一大目标就是鼓励有意求解这道难题的读者，希望你们能追随自己的见解和思路。

在写作本书的过程中，我有幸得到了许多人的帮助和支持。首先，我要感谢同事David Applegate、Robert Bixby和Vašek Chvátal。感谢我们在二十多年的岁月里，分享欢乐，共同工作，为揭开旅行商问题的部分奥秘而努力。

我还要感谢Michel Balinsky、Mark Baruch、Robert Bland、Sylvia Boyd、William Cunningham、Michel Goemans、Timothy Gowers、Nick Harvey、Keld Helsgaun、Alan Hoffman、David Johnson、Richard Karp、Mitchel Keller、Anton Kleywegt、Bernhard Korte、Harold Kuhn、Jan Karel Lenstra、George Nemhauser、Gary Parker、William Pulleyblank、Andre Rohe、Lex Schrijver、Bruce Shepherd、Stan Wagon、David Shmoys、Gerhard Woeginger和Phil Wolfe。感谢他们对旅行商问题及其历史的讨论。

书中使用的图片和史料来自Hernan Abeledo、Leonard Adleman、David Applegate、Masashi Aono、Jessie Brainerd、Robert Bixby、Adrian Bondy、Robert Bosch、John Bartholdi、Nicos Christofides、Sharlee Climer、James Dalgety、Todd Eckdahl、Daniel Espinoza、Greg Fasshauer、Lisa Fleischer、Philip Galanter、Brett Gibson、Marcos Goycoolea、Martin Grötschel、Merle Fulkerson Guthrie、Nick Harvey、Keld Helsgaun、Olaf Holland、Thomas Isrealsen、David Johnson、Michael Jünger、Brian Kernighan、Bärbel Klaaßen、Bernhard Korte、Drew Krause、Harold Kuhn、Pamela Walker Laird、Ailsa Land、Julian Lethbridge、Adam Letchford、Panagiotis Miliotis、J. Eric Morales、

Randall Munroe、Yuichi Nagata、Denis Naddef、Jaroslav Nešetřil、Manfred Padberg、Elias Pampalk、Rochelle Pluth、Ina Prinz、William Pulleyblank、Gerhard Reinelt、Giovanni Rinaldi、Ron Schreck、Éva Tardos、Mukund Thapa、Michael Trick、Marc Uetz、Yushi Uno、Günter Wallner、Jan Wiener和Uwe Zimmermann。感谢他们所有人的热情帮助。

两所院校为我提供了极好的写作环境，分别是佐治亚理工大学的米尔顿·斯图尔特工业与系统工程学院和普林斯顿大学的运筹学与金融工程系。我对旅行商问题的研究工作得到了美国国家科学基金会（CMMI-0726370）和美国海军研究办公室（N0014-09-1-0048）的资助，也收到了A. Russel Chandler III的慷慨捐赠。感谢他们一直以来的支持。

最后，我要感谢我的家人Monika、Benny和Linda。感谢他们多年来耐心听我讲述旅行商的故事。

第1章 难题大挑战

> 它产生于三个人求解一道经典数学问题的研究工作。这个历史悠久的问题叫做“旅行商问题”，无论靠人工计算还是借助最快的计算机都一直无法解决。
>
> ——IBM新闻稿，1964年[1]

1. 摘自IBM公司发布于1964年1月2日的新闻稿。“它”表示一个新的计算机程序，能够解决小规模的旅行商问题，由Michael Held、Richard Karp和Richard Shareshian三人编写完成。

1962年春天，宝洁公司发起了一场广告宣传活动，在应用数学家中引起了不小的反响。活动的重头戏是一项竞赛，奖金高达1万美元，在当时足以买下一座房子。参赛规则如下：

> 假设Toody和Muldoon打算开车环游美国，地图上用点标出的33个地点都要游览，并且要走满足条件的路线中最短的一条。请你为他们规划一条旅行路线，以伊利诺伊州的芝加哥市为旅途的起点和终点，依次用线连接各地点，并使得总里程最短。



图**1-1**　**“54**号车**”**竞赛题（宝洁公司供图）

Toody和Muldoon是当时一部美国热门电视剧[2]中的人物。他们是驾驶54号车的两名警官。这项游遍33座城市的任务是旅行商问题（traveling salesman problem，简称**TSP**）的一个具体例子。TSP的一般形式为：给定一组城市及它们两两之间的距离，求经过每座城市并返回出发地的最短路线。

2. 即1961～1963年播出的美国电视喜剧*Car 54, Where Are You*。——译者注

求解一般形式的TSP，是容易，还是困难，抑或无法求解？对此，最简单的回答就是谁也不知道。这道计算数学领域的知名难题之所以神秘莫测而又引人入胜，正是因为这一点。为此陷入困境的也不只是一名纠结的旅行商而已，因为在计算复杂度的本质与人类认识的可能限度这

一高深论题中，TSP正是讨论的焦点。

若你已跃跃欲试，那么只需要一支削尖的铅笔和一张干净的草稿纸，就可以向旅行商伸出援手。或许我们对于整个世界的认识也会因为你而发生飞跃。

1.1 环游美国之旅

TSP虽然公认棘手，但从某一方面来看却相当容易：经过给定一组城市的全部可能路线总数是有限的，因此1962年的某位数学家只需检验每条可能的路线，将最短的一条记录下来寄给宝洁公司，便可坐等一万美元支票寄到家中。这个解题策略堪称简单而完美，但有一点潜在的困难。由于路线总数极其庞大，根本不可能逐一检验。

1930年，奥地利数学家、经济学家Karl Menger已注意到TSP存在这种困难。最初正是因为他的工作，数学界才开始关注TSP这道难题。他写道："该问题当然可以在有限多次试验内解决，但是尚未发现能够给出比给定点的全排列数更低的试验次数的解法。"[1]一条路线可以通过到达各城市的顺序来唯一确定。我们依次用字母A～Z及数字1～7表示Toody和Muldoon的33个目的地，即A代表芝加哥市，B代表维奇托市，以此类推。如此，一条可能的路线便可以记作

ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567

或以上符号排成其他任一序列的形式。每一个这样的序列都是这33个符号的一个排列（permutation）。

1. Menger (1931).

由于旅行的起点和终点相同，每个排列对应的路线实际都是一条环形路线，在这条环形路线上选择另外一座城市作为起点就能得到另一个不同的序列，因此同一条环形路线对应33个不同的线性排列。为避免重复计数，不妨固定城市A为每条路线的起点，则第二座城市有32种可能选择，第三座城市有31种可能选择，以此类推。最后，可以得到环形路线的总数为32×31×30×…×3×2×1。这个数字记作**32!**，读作32的阶乘，表示32个不同物体总共有多少种排列情况，也称为全排列数。

在"54号车"竞赛题中，注意到从芝加哥到维奇托的距离等于从维奇托到芝加哥的距离，并且对于任意两座城市来说都是如此，所以我们可以减少一部分工作量。根据对称性，对于同一条路线，总路程与旅行方向无关。因此，字符串

ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567

和它的逆序字符串

7654321ZYXWVUTSRQPONMLKJIHGFEDCBA

虽然写法不同，但对应同一条环形路线。

这样一来，问题的可能路线数便可以减半，只剩下32!/2种排列等待

我们检验。先别急着拿笔，请注意，这个数字也就意味着我们要考察

131 565 418 466 846 765 083 609 006 080 000 000

条不同的路线。

现在，我们当然会把检验所有路线的任务都交给计算机完成。那么，让我们选一台大家伙——IBM“走鹊”（Roadrunner）超级计算机集群。它属于美国能源部，造价1.33亿美元，含有129 600个计算核心，运算速度可达每秒1457万亿次，高居2009年全球最快超级计算机五百强榜单首位[2]。假设检验每条路线只需要一次算术运算，那么用这台超级计算机解决含有33座城市的TSP，估计需要超过28 000亿年[3]，耗时漫长得可怕。要知道，宇宙从诞生至今也不过140亿年而已。难怪Menger对于TSP的暴力枚举解法并不满意。

2. 该榜单发布于2009年6月。但是在2012年11月的最新榜单上，Roadrunner已经退至第22位，而榜首的超级计算机Titan每秒运算速度已达到27 112万亿次。——译者注

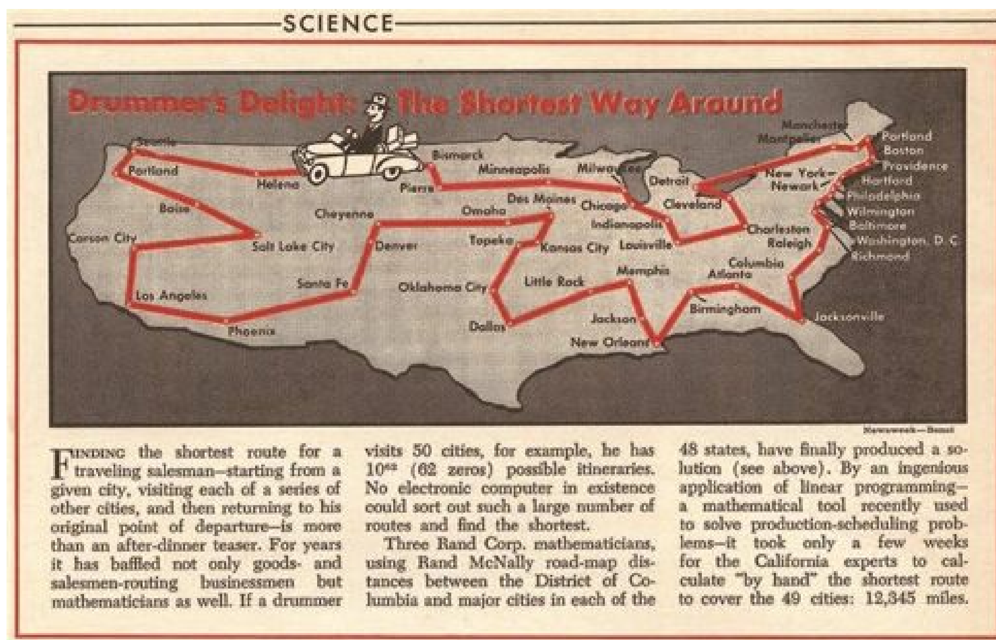
3. 原文为大约28万亿年（28 trillion），但实际计算结果应为28 600亿年（2.86 trillion）。——译者注

上述估算简短而引人思考，但是请务必牢记，Menger只是说更快的解法尚未发现，并没有否定它们存在的可能性。John Little等人[4]在宝洁公司的赛后评论里对此总结得很到位：“有些人可能学过太多知识，所以写信告诉宝洁公司该问题不可解。这种想法是对目前科技水平的一种很有意思的误解。”在评论中，Little等人还描述了在TSP的解法研究方面取得的突破。不过计算机代码改进得不够，对于33座城市的情形还无法真正解出答案。看起来，当时似乎全美国没有一个人有能力为Toody和Muldoon设计一条路线，并确保它就是满足条件的路线中最短的一条。

4. Little, J. D. C., et al. 1963. Operations Research 11, 972–989.

事实并非如此。尽管这道题确实是块难啃的骨头，但是若我们退回到1954年，就会发现，那时已经有一组人有这个能力。他们几乎一定能找到所求的最短路线，还能写出保证书一并寄去参赛。因为他们此前已经攻克了一道规模更大的类似问题，题目条件是环游美国，游览首府华盛顿和另外48座城市（分别属于当时美国的48个州）。20世纪30年代中期以来，这道难题在数学界内广为流传。《新闻周刊》报道了它的解决。[5]

5. *Newsweek*, July 26, 1954, page 74.



FINDING the shortest route for a traveling salesman—starting from a given city, visiting each of a series of other cities, and then returning to his original point of departure—is more than an after-dinner teaser. For years it has baffled not only goods- and salesmen-routing businessmen but mathematicians as well. If a drummer visits 50 cities, for example, he has 10^{62} (62 zeros) possible itineraries. No electronic computer in existence could sort out such a large number of routes and find the shortest.

Three Rand Corp. mathematicians, using Rand McNally road-map distances between the District of Columbia and major cities in each of the 48 states, have finally produced a solution (see above). By an ingenious application of linear programming—a mathematical tool recently used to solve production-scheduling problems—it took only a few weeks for the California experts to calculate "by hand" the shortest route to cover the 49 cities: 12,345 miles.

图**1-2**　推销员之喜，《新闻周刊》，**1954**年**7**月**26**日第**74**页

给定一组城市，有一名旅行商要从指定的某座城市出发，经过其他所有城市，最终返回出发地。为他寻找最短路线的问题，并不只是一道晚饭后的智力题。数年来，它不但难倒了规划运货路线和商旅路线的商人，而且让数学家也束手无策。举个例子，假如一个推销员要拜访50座城市，那他可选的路线就有10^{62}种，这个数字是1后面连着62个0。目前的任何一台电子计算机都无法在这么多路线中理清头绪，选出最短的那条。

在美国加利福尼亚州的兰德公司，三位数学家终于提供了一种解法。线性规划方法是最近用来解决生产计划管理问题的一种新的数学方法。他们巧妙地运用了这一方法，只用了几星期时间就通过“手工”计算得到了周游华盛顿和其他48座城市的最短路线，其长度为12 345英里（约19 867千米）。他们计算使用的各城市相距里程数据取自兰德麦克纳利道路地图（Rand McNally road-map）。

这三位数学家是George Dantzig、Ray Fulkerson和Selmer Johnson。他们所在的研究中心实力雄厚，影响力极大，专门研究数学规划这一新兴学科。该中心位于加利福尼亚州圣莫尼卡市，属于兰德公司。

在他们给出的保证书里，有一些运算过程完成得很漂亮，将在本书后面的章节进行讨论。现在，最好暂时把这份保证书理解为数学证明，和我们在几何课上学过的那种证明一样。首先，Dantzig等人证明了，在这道环游美国48州问题中，周游49座城市的路线长度不可能短于12 345

英里；然后，他们又提供了一条路线，其长度刚好等于12 345英里。论证与构造相结合，便彻底解决了这道题。

虽然他们没有参加那场奖金高达一万美元的广告竞赛，但是我们可以确定，按照他们的思路，用计算机解决33座城市的TSP已变得轻而易举。图1-3中画出了Toody和Muldoon的最短路线。回到1962年，尽管没有人能肯定这就是正确答案，然而确实有一些参赛者提交了这条路线，因此暂时并列第一，其中包括两位数学家Robert Karg和Gerald Thompson。他们两人发明了一种启发式试探策略[6]，从而找到了最短的路线。这个故事的结局很美好，至少对数学界来说是如此，因为决胜题要求写一篇短文赞美宝洁公司的某项产品，而Gerald Thompson凭借一篇赞颂肥皂的散文脱颖而出，最终赢得了大奖。

6. Karg, R. L., G. L. Thompson. 1964. Management Science 10, 225–248.



图**1-3**　**“54**号车**”**竞赛题的答案

1.2 不可能的任务吗

兰德小组的研究工作解决了环游美国48州的难题，却没有彻底解决TSP。他们取得了一次巨大的成功，并不意味着就能搞定规模更大的题目。事实上，如果拉斯维加斯赌场把TSP的最终结果作为一项赌博项目，那么在数学家看来，把赌注押在“TSP永远无法彻底解决”上会有更大的胜算。对此，必须明确一点：我们要找的所谓解法（solution），实际上是算法（algorithm），也就是要求对于任何一道实例，只要按照算法一步一步计算，一定能求出最优路线。单单找到周游美国或者周游其他某国的最优路线是不够的。

所以，为一般形式的TSP找到通用解法的难度可想而知。Charles Stross在科幻小说《抗体》（*Antibodies*）[1]中就写到了这一点。在这篇小说中，有人找到了旅行商问题的高效算法，结果就像世界末日一样，种种事件接踵而来。我们只能希望，TSP真相大白的那一刻不会宣告所谓的世界末日。无论如何，这件事一旦发生，必然会让世界天翻地覆。为什么？让我们先看看前人是怎么说的。

1. 英文版见G. Dozois编辑的“The Year's Best Science Fiction”（2000年度最佳科幻小说），2001年由St. Martin's Press出版。中文版刊登于《科幻世界》2010年7月刊第54~64页。——译者注

> 要想成功解决该问题，很可能需要另辟蹊径，使用前所未有的新方法。事实上，该问题的通用解法很可能压根不存在，若能证实这个结论也是很有价值的。

——Merrill Flood，1956年[2]

2. Flood, M. M. 1956. Operations Research 4, 61–75.

> 依我猜想，旅行商问题没有好的算法。

——Jack Edmonds，1967年[3]

3. Edmonds, J. 1967. J. Res. Nat. Bur. Stand. Sec. B 71, 233–240.

> 我们在本文中给出了一些定理。它们有力地暗示（尽管不足以证明）：该问题像许多别的问题一样，也将是一道永恒的难题。

——Richard Karp，1972年[4]

4. Karp (1972).

以上评论者是旅行商问题研究领域的三位大师。Merrill Flood在20世纪40年代呼吁人们支持研究TSP，并使它成为基础性研究课题，在这方

面，他的贡献无人能及。1956年，他在讨论该问题的研究现状时，首次提出高效解法根本不存在的可能性。10年后，Jack Edmonds重申这一看法，当时他和别人为通用解法是否存在而打赌，而他认为不存在。谈起自己这样想的根据，他谦虚地说："我对数学提出任何猜想，都是基于如下两条理由：第一，数学上有合理的可能性；第二，我并不确定。"不过，这只是他的玩笑话，因为对于20世纪的数学，他学识渊博，思考深邃，在世界上是数一数二的，所以他如此下注必然有他的道理。5年后，伟大的计算机科学家Richard Karp终于揭晓了这个赌的本质。他在一篇文章中把TSP和许多其他计算问题联系到了一起。具体理论细节留到第9章再讲，现在只稍做解释。相信这一节内容足以让读者理解，为何在小说《抗体》中，宣告发现TSP的高速算法会让每个人都不寒而栗。

1.2.1 好算法，坏算法

Edmonds在写下"好的算法"一词的时候，对于"好"的理解和我们一样：如果一个算法能够在我们满意的时间内解决问题，那么它就是好的。然而，他必须把"好"定义为正式的概念，才能让它有合理的数学意义。显然，我们不能指望TSP的每道题目都能在固定时间（比如一分钟）内通过人力或计算机解决，至少在城市数量增加时应该允许求解时间也相应增加。关键是要确定，时间按照什么速度增长才能得到我们的认可。[1]

1. 可能有人会说，考虑问题规模时，城市距离数据的精度也是影响因素。假如我们读取A城市和B城市之间的距离（或者旅行费用）时，需要读取几百万位数字，那么确实有影响。但是，即使每段路程都是小于常数K的整数，TSP依然相当困难，而这种复杂度是最重要的。我们关心的是TSP的本质复杂度，因此完全可以放心地忽略数据精度这样的细节。

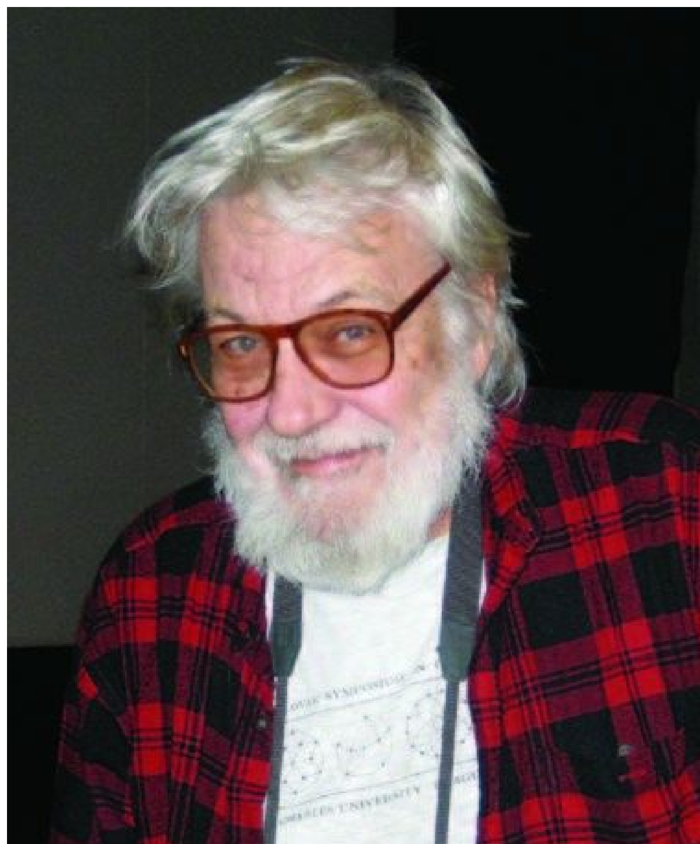


图**1-4**　**Jack Edmonds**（2009年摄，Marc Uetz提供）

表**1-1** 每秒运算$\mathbf{10^9}$次的计算机在不同条件下的运行时间

复杂度	n=10	n=25	n=50	n=100
n^3	0.000001秒	0.00002秒	0.0001秒	0.001秒
2^n	0.000001秒	0.03秒	13天	40万亿年

我们用字母n表示问题的规模，在TSP中就是城市的数量。由于读取目标城市列表所需的时间与n成正比，所以可能有些强势的老板就只给我们正比于n的解题时限。这样的人看问题过于乐观了。就连Edmonds本人都允许运行时间以更快的速度增加，用更明智的方式划分算法的好坏。好的算法能保证在至多正比于n^k的时间内完成运算，其中，指数k可以是任意值，比如取2、3或者更大的数，但必须是固定值，不能随着n的增加而增加。于是，算法的运行时间若是以n^3的速度增加，那么它就是好的，若是以n^n或2^n的速度增加，就是坏的。表1-1列出了每秒运算十亿次的计算机在n取不同值时相应的运算时间，以便让你感受到好坏之间的差距。可以看到，如果n=10，坏算法也够用了；但如果n取值

达到100左右，2^n阶的算法会算个没完没了，你肯定不希望出现这种结果。

Edmonds对"好"的正式定义未必总是符合我们的直觉。在一道100座城市的TSP面前，我们不会对需要执行n^{1000}步的算法感兴趣。尽管如此，Edmonds的想法还是彻底改变了计算数学。算法可以明确地分成"好"与"坏"两种类别，从此数学家有了确定的目标，对计算问题的兴趣也更加浓厚。在应用方面，一旦有人证明某道问题存在好的算法，研究人员便纷纷全力以赴，竞相寻找更低的指数k，通常能把运行时间界限降低到正比于n^2、n^3或n^4的程度，最终的计算机代码足以处理规模很大的问题。

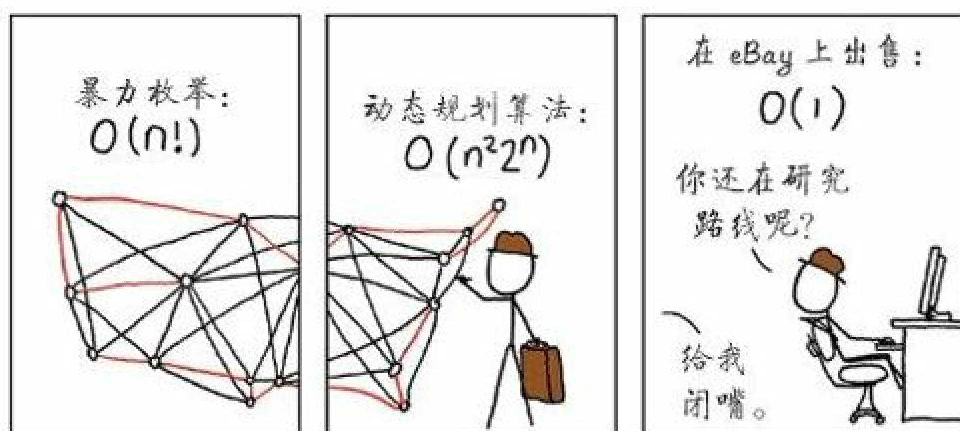


图**1-5**　旅行商问题（xkcd.com的Randall Munroe供图）

我要告诉TSP的爱好者一个遗憾的消息——TSP的好算法至今不为人知。迄今为止得到的最好结果发现于1962年，算法的运行时间正比于$n^2 2^n$。尽管这不是个好的算法，但是与经过n点的路线总数$(n-1)!/2$相比，运算时间增长得已经相当慢了，或许Menger的好奇心可以由此得到满足。

1.2.2 复杂度类P与NP

按照Edmonds划分算法的方式，计算问题也可同样分为两类，一类存在好的算法，另一类则不存在。我们喜欢第一类问题，将它们统称为**P**类。

为什么用字母P，而不用"G"（good）？这是因为研究人员认为"good"这个词带有主观情感，不太喜欢这种用法，所以多项式时间算

法（polynomial-time algorithm）就成为了标准术语。就是多项式（polynomial）。

P类问题的定义清晰明确，但是，有时却很难判断某道给定的问题是不是这一类“好的问题”。没准TSP也属于P类，只是我们还没找到好的算法来证明这一点而已。至少，我们看到最优路线时总能判断出来，这为我们保留了一线希望。事实上，假如我们面临的挑战是要找一条短于100英里的路线，那么，只要别人给我们一个答案，我们就可以轻松验证它的长度是否满足要求。由于这种性质，我们把TSP归为另一类问题，称为**NP**类。对于此类问题，我们总可以在多项式时间内验证某一个解是否正确。这里，两个字母NP来自非确定性多项式（non-deterministic polynomial）。“NP类”这个奇特的名称暂且抛开不提，这类问题本身是很自然的：我们提出计算需求时，理应有办法验证结果是否满足要求。

1.2.3 终极问题

P类和NP类有没有可能只是同一类问题的两个不同名字？有可能。1971年，Stephen Cook取得了突破性成果，提供了一种证明思路。（我们都姓Cook，但并没有亲戚关系，虽然我因为这样的误会享受过好几顿免费的大餐。）他的理论指出，NP类中存在一道题，只要我们能找到这一道NP题的好的算法，就能证明每一道NP题都存在好的算法。根据Cook、Karp和其他学者的工作，我们现在知道，这样的题其实不止一道。它们称为**NP**完全问题（NP-complete problem）。TSP就是最著名的NP完全问题。

若能对一个NP完全问题找到一个好算法，就足以证明P类和NP类是等价的问题类，即**P=NP**。因此，第一个找到TSP的通用解法的人，得到的财富将远远多于宝洁公司当年的大奖——克雷数学研究所悬赏100万美元，奖给能够证明或证否P=NP的人。

人们一般认为，P类和NP类并不等价，但是并没有充分的理论依据，只是感觉不应该奢求P=NP而已。因为P=NP的成立就意味着，只要能把一道题写成可以验证的形式，便立即得知它存在高效解法。而人们正是假设某些NP问题难以求解，并以此为基础建立了现行的密码体系，所以假如这些NP问题都有快速算法，那么网络贸易将陷入停滞。对于破解密码、入侵系统的黑客来说，这就像送给他们一把用来窃取数据的瑞士军刀一样。

与加密系统失败相比，小说《抗体》中的崩溃危机潜伏得更深。故事里，人工智能程序突然效率大增，从而推翻了有生命的主人并取而代之。但是，我们大概不用担心这些可怕的机器，P=NP带来的好处也可能远远大于不良后果。2009年，Lance Fortnow在一篇综述中写道："很多人只关注问题的负面，认为P=NP会让公开密钥加密技术完全失效。确实如此。但是P=NP的益处更大，足以使整个互联网变成历史上微不足道的脚注。"[1]他的理由是，最优化将因此变得轻而易举，旅行商能找到最短的路线，工厂能达到最大的生产能力，航班也能安排得当、避免延误，诸如此类问题都能得到解决。一言以蔽之，我们会更好地利用可用资源。科学界、经济界、工程界将出现更加强大的工具和方法，重大突破源源不断。接下来的数年内，诺贝尔奖委员会将忙得不可开交。那是一个如花似锦的美好世界，可是多数人都认为它不会实现。

1. Fortnow, L. 2009. Communications of the ACM 78, 78–86.

显然，解决P与NP问题是当代最重大的难题之一。在探讨TSP这样的NP完全问题的过程中，对于好的解法可能带来的种种后果，一定不要过分纠结。若不考虑其深远的影响，归结起来，TSP原本只是简单地为旅行商寻找路线而已。平凡与非凡只在天才的一念之间。

1.3 循序渐进，各个击破

将来或许会有人一举攻破终极的复杂性问题，给出惊天动地的答案。在那之前，我们能做什么？直面旅行商问题，目标非常明确：求解更大规模、更难解决的情形。

TSP是算法工程（algorithm engineering）[1]的代表问题。这门研究学科重视实用性，以不达目的誓不罢休为理念。理论上，TSP的规模一旦大到一定程度，求解某些实例的计算量就可能高得离谱。但这并不代表只要看到规模很大的具体问题，就只能退而求其次，选择粗略估计作为结果。研究人员正是在这种绝不妥协的态度的推动下，不断改进计算机代码和技术方法，如今已能解决复杂度惊人的大问题。

1. “算法工程”这个名词至少可以追溯到1997年，当年在意大利威尼斯举行了首届算法工程研讨会。一个课题组得到了德国科学基金会的资助，致力于算法工程的研究，并把这门学科的内容描述为“实用算法的设计、分析、实现与实验评估”。Petra Mutzel是课题领导者之一，同时也是多特蒙德大学的算法工程学教授。

在TSP研究领域，如果有人攻克了一道未解难题，消息会迅速流传开来，就像登顶喜马拉雅山脉某座处女峰或者打破百米短跑的纪录一样轰动。这并不是因为我们迫切渴望知道这道题目答案的具体细节，而是因为我们迫切需要知道TSP的研究能够继续推进，哪怕只是一小步。也许最终我们注定会败在旅行商手下，但是至少我们曾经拼搏过。

1.3.1 从49到85 900

兰德公司的Dantzig、Fulkerson和Johnson是TSP领域的三位英雄人物。虽然计算机时代已经拉开序幕，持续出现大批新人参与研究，但是Dantzig等人通过手工计算得到的49座城市的纪录始终无人能及。算法推陈出新，计算机代码反复编写，研究报告不断发表，然而年复一年，他们的纪录依然屹立不倒。终于，IBM研究员Michael Held和Richard Karp打破了Dantzig等人的垄断局面。Karp正是我们之前提到过的那位计算机科学家。他研究过TSP无法解决的可能性，但是显然并不满足于空谈理论。他们测试的范例是正方形区域内随机分布的64个点，以每两点之间的直线距离代表旅行费用。

接下来的好几年里，尽管有些研究小组对算法稍加改进，试图挤出一点提升的空间，但没有人能打破Held和Karp的绝对领先地位。直到

1975年，由Dantzig等人提出的方法重出江湖。这次，Panagiotis Miliotis对他们的想法做了一些改动，并计算出经过80个随机点的最短路线，使Held和Karp的纪录黯然失色。

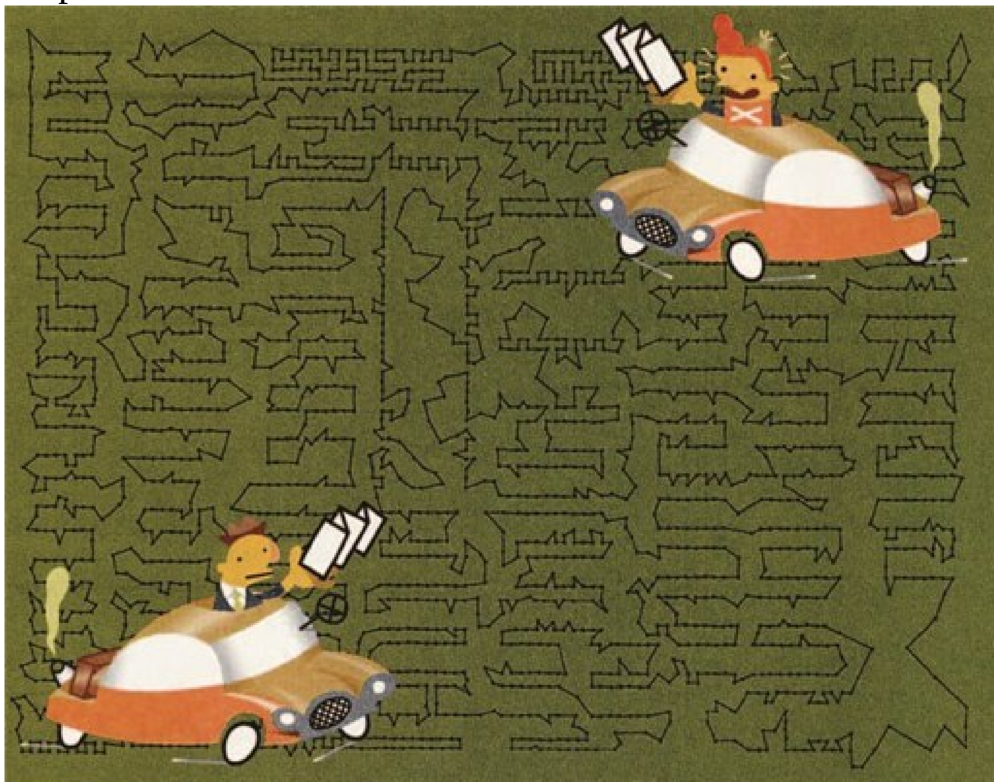


图**1-6　TSP**的新纪录，包含**3038**座城市，《发现》（***Discover***）杂志，**1993**年**1**月

由Miliotis的研究看来，Dantzig等人的方法可能具有极大的威力，远远超出人们对TSP计算极限的预期。此后不久，Martin Grötschel和Manfred Padberg所做的理论研究进一步证实了这一点。这两位数学家在TSP的舞台上叱咤风云长达15年之久，为TSP基本方法的重大推广奠定了基础。他们的成功始于1977年，当时Grötschel在博士论文中构造了周游德国120座城市的路线。随后，Padberg和IBM研究员Harlan Crowder合作，解决了一道出自电路板钻孔问题的实例，对318座城市的情形算出了结果。

尽管这两个结果本身都非常出色，但是直到1987年他们陆续宣布一系列惊人发现的时候，人们才发现，原来之前只是牛刀小试而已。1987年是TSP历史上的丰收年。Grötschel和Padberg两人的研究小组分别在大西洋两岸独立展开研究，很快解决了一道又一道实例：周游美国532座城市，环游世界666个地点，分别含有1002座城市和2392座城市的电路板钻孔问题……Grötschel当时在德国波恩大学和博士生Olaf Holland合作

研究，而Padberg则在美国纽约大学与意大利数学家Giovanni Rinaldi共事。

1988年初，乘着这股热潮，Vašek Chvátal和我也决定加入TSP计算的竞争中。前面有Grötschel和Holland、Padberg和Rinaldi的杰出工作，奋起直追的我们没有丝毫优势。不过我们很幸运，因为同时期有那么多来自世界各地的研究人员，他们非常活跃，对该问题的理论方面开展了前所未有的深入钻研。我们仅仅通过筛一遍日益增加的相关研究，就能获取可以用来解决计算问题的强大工具。不过，在正式动手之前，我们迈出了所有工作中最重要的一步——招收David Applegate和Robert Bixby为小组成员，他们两位是当代实力最强的计算数学家。研究起初进度缓慢，好几次走错了方向。1992年，我们成功利用计算机网络并行计算，解决了一道来自电路板钻孔的问题，其规模为3038座城市，创下当时纪录。此时，方法已经成形。在此基础上，我们又在1998年计算得出了周游美国13 509座城市的最优路线，于2004年得到了周游瑞典24 978座城市的最优路线，最终在2006年解决了一道来自实际应用的问题，其规模达到85 900座城市。我们使用的计算机代码名为“Concorde”[1]，网上到处都可以找得到。

1. Concorde代码是用C语言编写的，源代码和程序都公开提供下载。读者如果想深入学习，可以访问http://www.tsp.gatech.edu/concorde/index.html获得相关资源。——译者注

在最后一道创纪录的问题里，85 900座城市代表的是一枚计算机芯片上的不同位点。为了加工定制芯片，需要用激光切割这些位点，而激光光头在各点之间的移动顺序便可转化为TSP模型。尽管每次移动距离只是毫米量级，但总移动时间对芯片生产成本的影响却很大。图1-7和图1-8分别是激光移动的最优路线图和路线局部放大图。

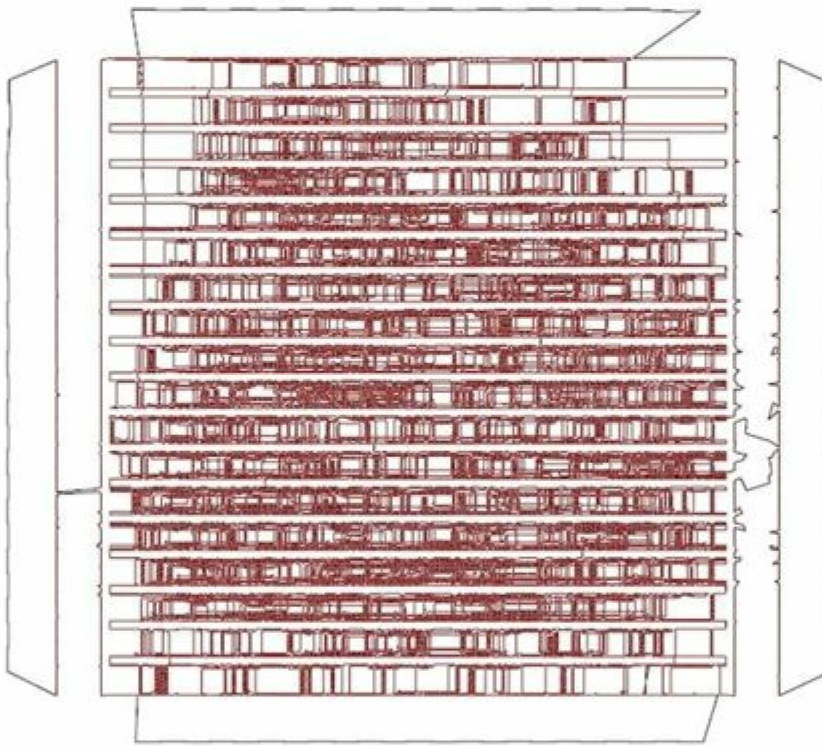


图**1-7**　包含**85 900**座城市的**TSP**路线，题目出自计算机芯片应用

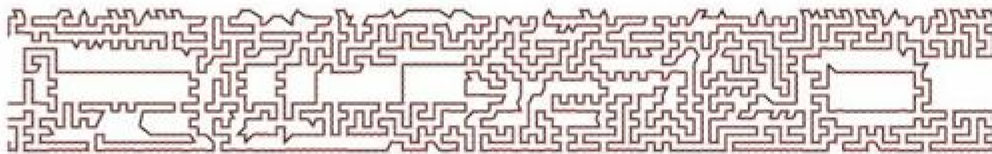


图**1-8**　包含**85 900**座城市的**TSP**路线局部放大图

1.3.2 世界旅行商问题

在包含85 900座城市的芯片问题和其他一些电路板钻孔的应用实例中，各点都具有类似网格点的分布形式。早年环游美国48州的题目开启了TSP的漫长研究，而这些例子显然没有真正体现当初那种旅行的精神。不过，仔细观察图1-9所示的三条周游德国的路线，还是很容易体会到，现代的结果在复杂度上的确有所提高。图中规模最小的路线经过33座城市，简称为《指南手册》路线，出自1832年的一本旅行商指南小册子。蓝色路线经过120座城市，是当年Grötschel破纪录的成果。背景里的那条路线则是2001年由Concorde代码对15 112座城市计算得到的最优路线。

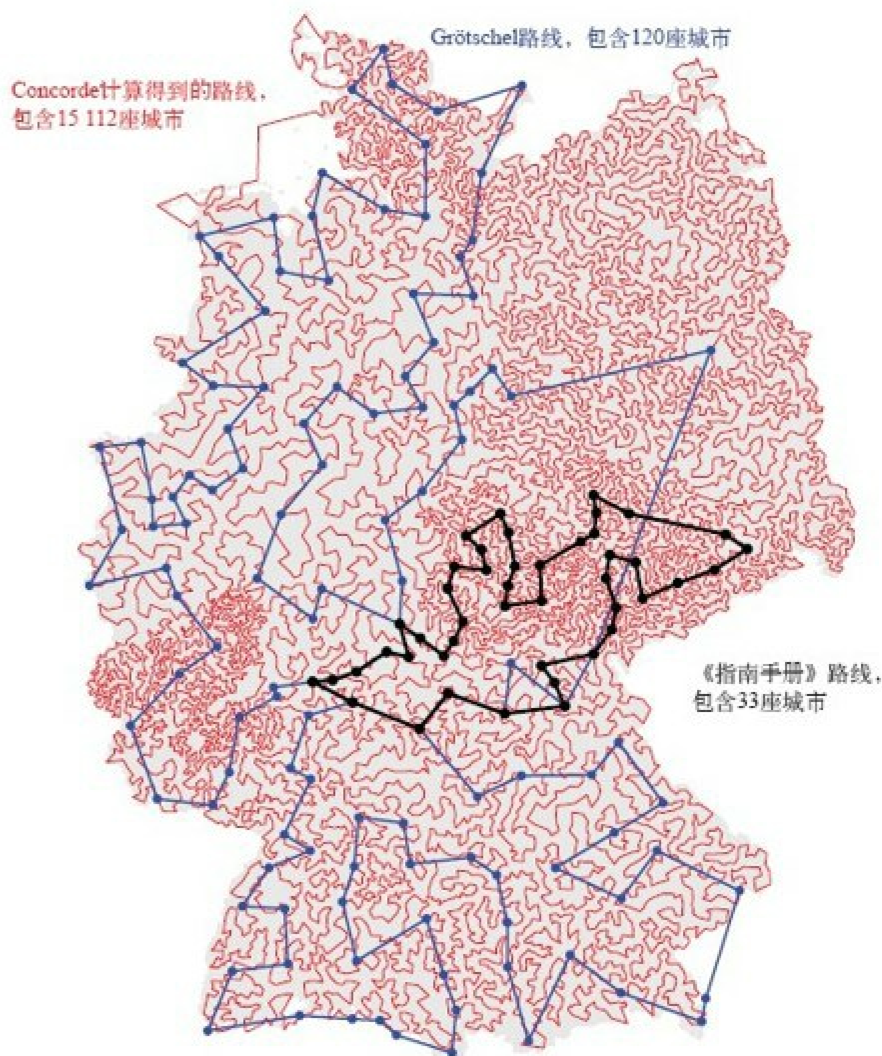


图**1-9**　周游德国的三条路线

也许对德国之旅来说，这条包含15 112座城市的路线就是最完满的路线了。但如果我们将世界上所有城市、区县、乡镇都囊括在内，就连南极的几处科考基地也算上，构造出一个总共包含1 904 711座“城市”的终极旅行挑战，这条路线便远远不够了。这道世界旅行商问题诞生于2001年。来自世界各地的计算机代码纷纷在它面前败下阵来，Concorde也不例外。如果克雷数学研究所的百万悬赏不合你胃口，或许这道世界级难题能点燃你的斗志。截至本书出版之时，本题最著名的路线是由丹麦计算机科学家Keld Helsgaun于2010年10月10日发现的，长度为7 515 790 345米。我们几乎可以肯定，这并非最好的结果，但也已由Concorde计算得知，不存在长度短于7 512 218 268米的路线。这样一来，Helsgaun给出的解最多只比真正的最短路线长0.0476%。两者已经很接近，不过依然有充足的余地让我们再抄几条近道。

1.3.3 《蒙娜丽莎》一笔画

为上面那道世界旅行商问题找到最优路线将是了不起的成就，但是求解需要用到的工具方法很可能要在10年以后才会出现。幸好，在征服世界的途中，从来不缺有趣的问题。一个漂亮的例子就是如图1-10所示的《蒙娜丽莎》TSP，总共包含100 000座城市。

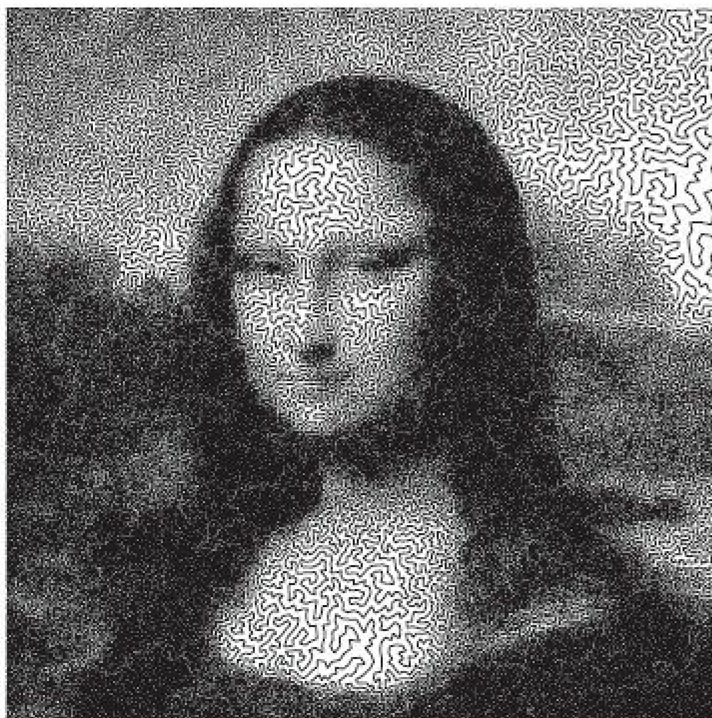


图**1-10**　根据列奥纳多·达·芬奇的画作《蒙娜丽莎》设计的**TSP**，路线由永田裕一发现

Robert Bosch于2009年2月提出了这道题目的数据集，希望用一条连续曲线画成达·芬奇的这幅名画。日本北陆先端科学技术大学院大学（JAIST）的永田裕一（Yuichi Nagata）取得了迄今为止最好的结果。他的路线最多只比最优路线长0.003%。差距如此之小，令人迫不及待，但毕竟还没有真正完成。有一项1000美元的奖金将颁发给第一个突破永田裕一结果的人，以激励有意参与这道题目的研究者。奖金不错，但是请你铭记在心：问题挑战之所以一道接一道，真正的目的在于积累思路，以便在TSP的通用解法研究以及更大范围的应用领域派上用场。解决问题的新途径才是关键所在。

1.4 本书路线一览

图1-11的T恤衫图案出自Jessie Brainerd之手，她曾参加2007年的布达佩斯数学项目。[1]在应用数学专业或者计算机科学专业，每一名毕业不久的合格本科生都能一眼看出这幅图的主题就是TSP。按照许多大学的课程设置，学习旅行商问题都是必经之路。最近，甚至连美国的中学课本里都有该问题的简单介绍。



图**1-11　2007**年万圣节的**TSP**（照片由Jessie Brainerd提供）

[1] 照片上身着T恤衫的男生是Bill Kay。他是Jessie Brainerd的同学，当时他们在布达佩斯参加万圣节晚会。Jessie Brainerd在博客上写了一篇日志，提到晚会上还有两名学生化装成P和NP，手持飞镖枪玩具互相打斗。

对TSP的介绍既然已经如此普及，我写作本书的目的何在？很简单，我希望能让本书读者对它更加了解，不止步于最简单的认识，而是远远深入问题，接触最近的理论进展与最先进的解法。我的终极目标是鼓励你们，希望你们也能踏上对旅行商的追踪之路。也许就在某个未知

的拐角处，你们将迈出最终一步，迎来彻底的成功。

本书第2章将从数学和应用两方面追溯旅行商问题的由来，在回顾历史的过程中介绍TSP的基本研究题目，进一步讨论将留到后面各章进行。第3章继续介绍TSP的丰富应用，包括旅行规划、基因组测序、行星搜寻、乐曲编排等方面的内容。

第4章到第7章阐述求解TSP的技术方法，为核心内容。接下来在第8章讨论计算机代码如何解决大规模的TSP题目。

第9章将探讨一般形式的TSP是否存在多项式时间解法。这一理论问题价值百万美金，你若爱钱，这章内容正合你意。然而，即使你钱库空空，亟待现金入账，我也不建议你跳过前面的章节。因为在计算领域大显身手的方法里，很可能孕育着成功解决理论问题的种子。而你若打算证明不存在通用解法，也必须解释清楚为何前人的解法能实际应用却不合要求。

数学知识就介绍到这里。第10章将转向心理学和神经学的研究范畴，论述人类如何在不借助计算机的情况下求解TSP。第11章将带领读者欣赏艺术作品中的TSP路线，比如Julian Lethbridge笔下美妙的抽象画和Robert Bosch设计的简单曲线图案。最后，第12章将再次号召本书读者接受TSP的挑战，并以此结束全书。

不屈不挠，前进到底

补充一点小建议。

如果面前有无数艰难险阻横空而降，爱尔兰作家J. P. Donleavy笔下的人物Rashers Ronald会发誓“不屈不挠，前进到底”。[1]Applegate等人在研究TSP计算时便将这句话作为战斗口号。我建议深入该问题的读者也能坚持这种态度。本书会介绍许多专家，他们的研究取得了重大进展，但仍然没有从根本上解决TSP。要想扭转局势，搞定旅行商，也许我们需要的恰恰只是一个新的思考角度。

1 Rashers Ronald出现在J. P. Donleavy的作品*The Destinies of Darcy Dancer, Gentleman*中，该书由Atlantic Monthly Press于1994年出版。



图**1-12**　左图：**1987**年，**W. Cook**（本书作者，左一）和**V. Chvátal**（右一）向作家**J. P. Donleavy**赠送夜壶，由**Adrian Bondy**拍摄，照片所有权利保留；右图：**1987**年，**J. P. Donleavy**写给本书作者的明信片

第2章 历史渊源

> 多年来，数学家似乎一直都在数学会议上讨论这个问题，尽管它并不是正式议题。
>
> ——George Dantzig，Ray Fulkerson，Selmer Johnson，1954年[1]

1. Dantzig, Fulkerson, and Johnson (1954).

旅行商问题流传甚广，在数学界颇负盛名，但是它发展成名的历程却迷蒙不清，鲜为人知。就连“旅行商问题”这个生动的名字究竟启用于哪个年代，我们都无法确定。即便如此，我们依然可以讲述这段故事的大部分内容，只不过时常要做一些合理的推测。TSP的历史故事可以作为铺垫，让我们在继续深入该问题之前，首先具备基本的认识和思考，从而更好地了解近年的具体研究工作。

2.1 数学家出场之前

人类早就开始着手解决具体的TSP应用问题，而关于它的数学研究在很久以后才渐渐蔚然成风。无疑早在穴居人的年代，我们的祖先就在狩猎采集时解决了一些路线规划问题。这些规划问题的规模很小，可能不像长期规划那样有意义。然而，最近几百年来，在某些特定行业，精心规划的路线显然对从业人员很有帮助。我们的讨论不妨就从观察这些人的旅途开始。

2.1.1 商人

说起旅行商问题的由来，须知旅行商确实是最早开始规划路线的人。图2-1是一张城市列表，来自旅行商H. M. Cleveland在1925年写的书信。[1]Cleveland在美国培基种子公司（Page Seed Company）上班，工作内容是收集玉米等农作物的订单。这张表摘自一份文件，全文共5页，列出了他在缅因州巡回的主要目的地。差旅从7月9日开始，一直持续到8月24日才结束，全程共经过350站，数目相当惊人。

1. 这张表来自本书作者从eBay购买的一组藏品。这批收藏品数量很大，都是H. M. Cleveland寄给培基种子公司的账单和信件，能买到它们实在是运气不错。但我并不是总能买到有意思的东西，比如我还买过某个旅行商写了50年的日记，结果发现，他的“旅行”只限于纽约州雪城周围的五六座城市而已。

图**2-1**　培基种子公司旅行商的缅因州目的地列表（全文共**5**页），**1925**年

显然，Cleveland和培基种子公司当时很关心如何才能让旅途时间最短。我们有两个证据：第一，如图2-2所示，他的旅行路线明显效率很高，虽然有一些回头路，但都是限于当时的道路建设情况，个别城镇与外界之间只有一条路相通，所以只能选择原路返回；第二，请仔细阅读下面这封Cleveland写给其老板的信。

> 尊敬的先生：
>
> 我的路线乱得一团糟，从没见过这么乱的。又要花掉去年一半的工夫来整理路线了。我已经改了一部分，从斯托克顿斯普林斯开始，挨个经过法兰克福、温特波特、汉普登海兰兹、班戈、斯蒂尔沃特、奥罗诺、奥尔德敦、米尔福德、布拉德利、布鲁尔、南布鲁尔、奥灵顿、南奥灵顿、巴克斯波特，然后到了奥兰，再接着走原来的路线。
>
> 希望您能将我1924年用过的单子寄给我，我想要从德克斯特开始的后半程路线。那个路线比手头这个好太多了。我不明白您到底为什么要让路线有那么大的跳跃，特别是从阿尔比恩居然一下子跳到麦迪

> 逊，简直是从地图这一头跳到那一头。这部分我自己改了。
>
> 从班戈开始，往南就没有桥能过河了，可您还把那里的城镇都列在单子上，就好像我有本事过去似的。上一季列出来的单子是最好的，我实在看不出为什么要改。我想我已经把话都说清楚了。
>
> 您忠诚的H. M. Cleveland
>
> 1925年7月15日

可见，Cleveland对部分路线非常不满，并且自己动手改进路线的设计。

图**2-2**　培基种子公司旅行商的缅因州差旅路线图，**1925**年。旅行从基特里出发，最终回到斯普林韦尔。两座城市距离很近，均位于缅因州南部

1925年，Cleveland去了很多地方，缅因州只是目的地之一。他的旅途还经过了康涅狄格州、马萨诸塞州、纽约州和佛蒙特州，总共到了一千多个目的地。他也绝不是唯一一个来回奔波的人。Timonthy Spear为美国旅行商写了一本书，名为*100 Years on the Road: The Traveling Salesman in American Culture*（《百年奔波：美国文化中的旅行

商》）。他在书中提到，据*Commercial Travelers Magazine*（《旅行推销员杂志》）估计，1883年美国有20万旅行商；到了19世纪末20世纪初，估计有35万人；20世纪早期，这个数字仍在增长；到了Cleveland的年代，旅行商已经是大多数美国乡镇的常客了。

Spear也描述了这些旅行商如何参考L. P. Brockett的*Commercial Traveller's Guide Book*（《旅行推销员指导手册》）之类的工具书，在自己工作的区域内设计路线。然而，多数情况下，路线是由总部预先制定好的，就像培基种子公司的做法一样。一种优化方法如图2-3所示，工作人员在地图上用图钉和细绳标出各条备选路线，并借此比较路线长度。



图**2-3**　兰德·麦克纳利的地图收纳柜与标针地图，《秘书学》（***Secretarial Studies***），**1922**年

一本1832年出版的德国手册是这段故事的重要参考文献，这本书德语书名为*Der Handlungsreisende — Von einem alten Commis-Voyageur*，翻译成中文就是《老推销员写给新旅行商的指南手册》[2]。简便起见，以后都称其为《指南手册》。手册里提到，好的路线很有必要。[3]

2. *Der Handlungsreisende — wie er sein soll und was er zu thun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiss zu sein — Von einem alten Commis-Voyageur.* B. Fr. Voigt, Ilmenau, 1832. 完整书名意为《老推销员写给新旅行商的指南手册：你该如何表现、如何工作、如何确保成功》

3. 该书节选片段由Linda Cook从德文原文翻译为英文，后转译为中文。——译者注

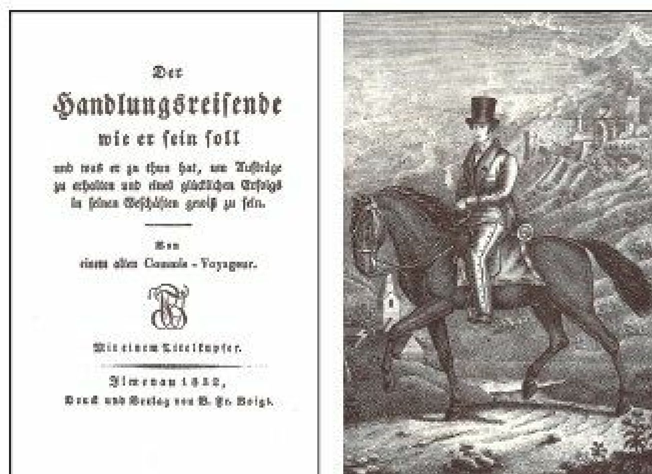


图**2-4**　德国旅行商指南手册，**1832**年

> 由于商务需求，旅行商四处旅行。放之四海而皆准的好路线并不存在，但是通过有效的选取和分解路线，可以节省大量旅行时间，因此我们觉得有义务提供一些这方面的指导意见。只要这些建议对你们有帮助，你们就应当尽量采纳和实践。我们可以保证，对于穿行德国的旅途，由于距离遥远，再加上来回奔波（请格外注意这一点），其他路线都不可能比我们的做法更节约。要记住，重点是到达尽可能多的地点，同时避免重复经过同一个地点。

这里明确叙述了旅行商问题，而叙述者本人就是一名旅行商！

《指南手册》提供了经过德国和瑞士境内不同地区的5条路线。在其中4条路线里，都出现了作为局部根据地的城市，因此这些城市都要到达不止一次。但是第5条路线如图2-5所示，是一条真正的旅行商路线，其旅行区域在德国所处的位置参见图1-9。正如《指南手册》所说，这条路线非常好，考虑到当时的道路情况，它甚至可能是最优路线。

若以为推销员的生活
没有艰难困苦与风波，
你必然大错特错。
因为他在泥泞中跋涉，
冒着雨雪和冰雹工作。
他拎起提包，勇敢上路，
走南闯北寻顾客。

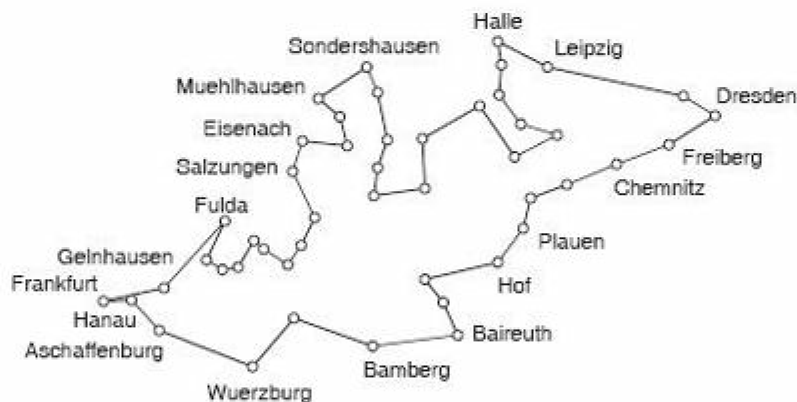


图**2-5** 《指南手册》提供的路线，**1832**年

19世纪中后期，人们精挑细选美国、英国等国家的旅行路线，集结成册出版了大量书籍。旅行商的形象充满浪漫色彩，也在戏剧、电影、文学与歌曲中留下了身影。在19和20世纪之交的旅行商诗作中，下面这段很有代表性。它摘自一本作品汇编集，出版于1892年。[4]

> 若以为推销员的生活
> 没有艰难困苦与风波，
> 你必然大错特错。
> 因为他在泥泞中跋涉，
> 冒着雨雪和冰雹工作。
> 他拎起提包，勇敢上路，
> 走南闯北寻顾客。

4. 这段诗是诗歌The Drummer（《推销员》）的开头部分。作品集：Marshall, G. L. 1892. O'er Rail and Cross-ties with Gripsack. *A Compilation on the Commercial Traveler*. New York, G. W. Dillingham。

推销员的辛苦工作与寻找路线的任务甚至出现在桌面游戏中。1890年，麦克劳克林兄弟公司（McLoughlin Brothers）发明了“旅行推销员游戏”（Commercial Traveller），玩家要在铁路系统里自己建造路线。

选择旅行商作为TSP的代言人，显然有理有据。

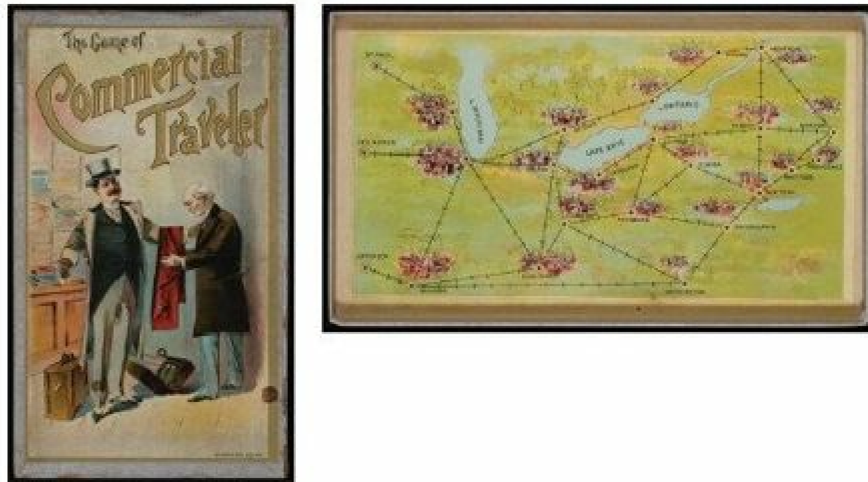


图**2-6**　旅行推销员游戏，由麦克劳克林兄弟公司于**1890**年推出，**Pamela Walker Laird**供图

2.1.2 律师

虽然最先出场的是旅行商，但是也曾有一些从事其他职业的人需要四处奔波。早在15世纪，《牛津英语词典》里就收录了“circuit”一词，意为巡回旅行或巡回路线。这个词的由来与英国的司法管辖区制度有关。当时英国各地的法庭是每年定期开庭审理案件的，因此法官和律师都要在自己管辖的各城镇之间巡回。后来，美国也采用了这种制度。时至今日，虽然法官不再需要巡回奔波了，但美国人仍然把地方法庭称为巡回法庭。

美国历史上最著名的巡回律师很可能是青年时代的亚伯拉罕·林肯。林肯在成为美国第16任总统之前，曾经在法律界工作，参与伊利诺伊州第八法庭区的巡回，负责14个县的公务。Guy Fraker描述了林肯的旅途。[1]

1. Fraker, G. C. 2004. Journal of the Abraham Lincoln Association 25, 76-97.

> 每年春秋两季，法庭都会连续开审数周，依次在14个县[2]分别开庭。除了斯普林菲尔德（Springfield）以外，其他每个县开庭最多一周。斯普林菲尔德是桑加蒙县的首府，也是伊利诺伊州的首府。秋季法庭在这里开庭两周。然后所有律师动身前往55英里以外的北京（Pekin），这里于1850年取代特里蒙特成为了塔兹韦尔县的首府。一周以后，他们再行进35英里，在麦特毛拉（Metamora）待三天时

> 间。接下来向东南方向前进30英里，到达布鲁明顿（Bloomington）。这是旅途中的第二大城市，因此事务较多，他们可能要在此地多逗留几日。下一站是相距35英里的普拉斯基山（Mt. Pulaski），这里于1848年取代波斯特维尔成为了洛根县的首府，几年后又将被新成立的林肯市取代——但是此时林肯尚是巡回律师中的一员。众人继续赶赴下一个县，然后是另一个，目的地一个接一个，直到走完整条巡回路线。全程一共经过11周时间，总路程超过400英里。

2. 美国政治区划制度与我国不同。虽然“county”翻译为“县”（有时翻译为“郡”），但这一级行政区仅次于州，不应与我国的“县”混淆。——译者注

文中，Fraker还提到，多次走完巡回全程的法庭工作人员为数极少，林肯是其中之一。图2-7中画出了1850年林肯和其他工作人员的巡回路线。这跟最短路线还有一些差距，至少只考虑距离、不考虑路况的话还可以更短。但是可以明显看出，规划路线的人确实有意尽量缩短巡回路线的长度。



图**2-7**　**1850**年伊利诺伊州第八法庭区的巡回路线

2.1.3 牧师

也许是法官和律师的差旅带来了巡回一词，但是若论巡回工作的名气，18世纪和19世纪的基督教牧师毫不逊色。约翰·卫斯理（John Wesley）是基督教新教卫斯理宗[1]的创立者。1791年，John Hampson在卫斯理的传记中写道："在英国和美国，每个地区都分成固定的几部分。这些小区域称为巡回区域，一般包含20～30个地点，由巡回牧师负责。巡回牧师人数一般在二到四人之间，每巡回一次需要花费一个月或者一个半月的时间。"[2]在英国、美国和加拿大的民间，都留下了关于这些巡回牧师的旅行情况的传说。你可以从下面这些摘录里体会到他们的路程有多长。

1. 又称为循道宗、卫理宗，是基督教新教主要宗派之一。——译者注

2. Hampson, J. 1791. *Memoirs of the late Rev. John Wesley*. J. Johnson, London, UK.

> 我路上经过了大约5000英里，向人们做了大约500场布道，走过了弗吉尼亚州和北卡罗来纳州的大部分路线。
>
> ——Freeborn Garrettson，1781年[3]

3. Banks, N. 1830. *The Life of the Rev. Freeborn Garrettson: Compiled from his Printed and Manuscript Journals, and other Authentic Documents*. New York. Published by J. Emory and R. Waugh.

> 当时我们的巡回路线全程500英里，我在短短四周之内做了63场布道，赶了500英里路。这实在太难了，但是我向上帝呼喊，上帝听到了我的祈求，赐予了我强大的力量。
>
> ——Billy Hibbard，1825年[4]

4. Hibbard, B. 1825. *Memoirs of the Life and Travels of B. Hibbard: Minister of the Gospel, Containing an Account of his Experience of Religion*. New York. Published by the author.

他们都是卫理公会教派的牧师，我没有资料，不知道较长的巡回传道路线具体是怎么走的。但是完全可以推测，他们选择的路线肯定也经过了一定的规划。对他们来说，工作的一大目标就是接触尽可能多的教友，因此缩短旅途中的时间肯定是个重要的考虑因素。

2.2 欧拉和哈密顿

那个时代，在数学的王国里，数学家正在为飞速发展的自然科学打下根基，研究工作忙得不可开交，无暇顾及旅行商、巡回律师和牧师面临的难题。然而，数学界的两大领军人物确实曾经探索过旅行商问题的部分领域。如今，我们把他们两人推崇为TSP研究的开山鼻祖。这是他们应得的评价。

2.2.1 图论与哥尼斯堡七桥问题

有关路线问题的早期数学文献中，最重要的一篇论文来自伟大的莱昂哈德·欧拉（Leonhard Euler）。The Euler Archive（欧拉档案网站）引用了历史学家Clifford Truesdell的一句评价："若能列出18世纪数学、物理学、工程学、天文学、航海学方面的所有论文，则其中欧拉的作品将占到整整四分之一。"欧拉是历史上最多产的数学家。他的研究范围很广泛，其中有一道难题曾经长期困扰哥尼斯堡的居民。

这座城市旧称哥尼斯堡，当时属于东普鲁士，如今已改名为加里宁格勒。图2-8是当地的卫星图像。可以看到，普莱格尔河流经哥尼斯堡，形成了错综复杂的水路。中间的长方形岛屿是克奈芳福岛（Kneiphof），河道在岛的两侧分流而过；东侧是较大的朗兹岛（Lomse），两岛隔河相望；河流北岸的部分为老城区（Altstadt），南侧则是郊区（Vorstadt）。[1]

1. Gribkovskaia, I., O. Halskau, G. Laport. 2007. Networks 49, 199–203.



图**2-8**　哥尼斯堡（今加里宁格勒）和普莱格尔河（今普列戈利亚河），**TerraServer.com**，**2011**年

在欧拉那个年代，普莱格尔河上一共架有七座桥梁：克奈芳福岛和老城区之间有绿桥（Green）和下桥（Köttel）两座桥梁，克奈芳福岛和郊区之间也有商人桥（Krämer）和铁匠桥（Schmiede）两座桥梁，克奈芳福岛和朗兹岛之间是蜜桥（Honey），朗兹岛和老城区之间是高桥（High），朗兹岛和郊区之间则由木桥（Wood）连接。哥尼斯堡的居民都是好市民，喜欢在城里散步，通过绿桥、下桥、商人桥、铁匠桥、蜜桥、高桥[3]、木桥这七座桥往返于普莱格尔河两岸。据说，当地流传着一道多年未解的难题：如何在一场散步中走过全城各地区，七座桥中的每一座都必须恰好走过一次？

3. 原文误作Lomse，实际应为High。——译者注

这就是哥尼斯堡七桥问题（Königsberg problem）。1735年8月26日，欧拉向圣彼得堡科学院提交了一篇论文，发表了对该问题的看法。[4]为了抓住问题的本质，他按照经典的数学思想，只把必要的信息提取出来，从而解决了问题，也开辟出了一个新的数学分支。这个新分支就是图论（graph theory）。[5]

4. Euler, E. 1741. Comm. academiae scientiarum Petropolitanae 8, 128–140.

5. Biggs等人将欧拉的论文译成了英文，并对其影响做了详细的学术论述，参见*Graph Theory 一书。

首先，欧拉抛开这个问题的具体背景，将城镇、河流、桥梁抽象出来，画成示意图，如图2-9所示。（图2-9取自欧拉原始论文副本的扫描

件，图片经过处理。）他分别用大写字母A、B、C、D表示哥尼斯堡城的4个区域，又用小写字母a～g表示普莱格尔河上的七座桥。这11个字母足以描述城里的任何一条路线，比如一条路线可以描述为：从A地经过c桥到达C地，再从C地经过g桥到达D地，从D地经过f桥到达B地，最后从B地经过b桥回到A地。这条路线也可以简记为AcCgDfBbA。欧拉的论证完全建立在这种表示的基础上。他把路线视为字符串，而不是行人的实际散步路线。

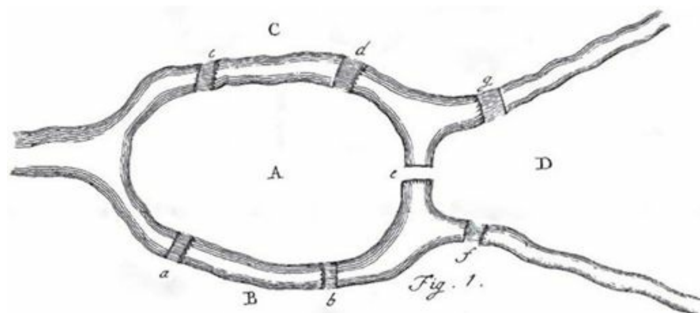


图**2-9**　欧拉简化后的哥尼斯堡七桥图

在这道题目里，A、B、C、D 4个区域的面积与论证完全无关，因此可以把它们都抽象成一点，而把a到g的桥梁抽象成连接两点的线，从而把题目形象化，画成一张更简单的图，如图2-10所示。图的含义不受点和线的形状、大小的影响，只和线的连接方式有关。这就是图论中的图（graph）。其中，点称为顶点（vertex），线称为边（edge），每条边两端的点是边的两个端点（end）。

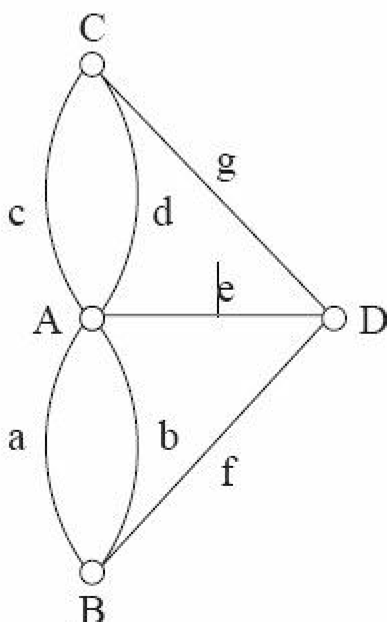


图**2-10**　哥尼斯堡七桥问题的图表示

经过简化以后，居民在哥尼斯堡城散步便可以理解成沿着边在图的各顶点之间移动。从B到D的一条散步路线可以写成BaAcCgDeAbBfD。在这条路线经过的边中，与顶点B相连的边有3条，分别是a、b、f；与顶点A相连的边有4条，分别是a、c、e、b；与顶点C相连的边有2条，分别是c、g；与顶点D相连的边有3条，分别是g、e、f。经过各顶点的边数依次为奇—偶—偶—奇，这样的排列并非出于偶然。欧拉发现了最关键的一点：若散步的起点和终点不同，则路线中，分别以起点或终点为端点的边都是奇数条，以途中任何一点为端点的边则都是偶数条；此外，如果散步路线是闭合的，也就是说起点和终点是同一点，那么以任何一个顶点为端点的边都有偶数条。综上可见，要么所有顶点都是“偶点”，要么恰好有2个顶点是“奇点”。

对哥尼斯堡城的居民们来说，这真是个坏消息。七桥问题的图里，4个顶点都与奇数条边相连，都是“奇点”，所以不可能在一次散步中经过每座桥恰好一次。欧拉的论证非常简短，却结束了哥尼斯堡问题的所有争论。

2.2.2 骑士周游问题

几年后，欧拉又写了一篇论文，再次讨论周游路线难题。[1]这次的题目来自国际象棋，称为骑士周游问题（knight's tour problem）。问题是：能否按照一定顺序移动骑士，让骑士从棋盘上的某一格出发，走遍整张棋盘，落在每一格上恰好一次，最后回到出发的格子？[2]欧拉的解法如图2-11所示，每格的数字表示骑士出发后第几步到达这一格。

1. Euler, L. 1766. Mémoires de l'Academie Royale des Sciences et Belles Lettres, Année 1759, Berlin. 310–337.

2. 国际象棋的标准棋盘有8行8列，骑士相当于中国象棋中的马，走“日”字格，即每移动一步时，或者横走两格竖走一格，或者横走一格竖走两格。——译者注

42	57	44	9	40	21	46	7
55	10	41	58	45	8	39	20
12	43	56	61	22	59	6	47
63	54	11	30	25	28	19	38
32	13	62	27	60	23	48	5
53	64	31	24	29	26	37	18
14	33	2	51	16	35	4	49
1	52	15	34	3	50	17	36

图**2-11**　欧拉为骑士周游问题给出的解法

欧拉对骑士周游这种想法很感兴趣，对于棋盘不是标准尺寸的情形也给出了解法。这些问题可以用图清楚地表示出来。我们用顶点表示棋盘上的格子，若骑士只用一步就可以从某格走到另一格，则在相应的两点间连一条边。骑士周游路线是指一条闭合路线，它到达每个顶点刚好一次。（在哥尼斯堡七桥问题里，所求的散步路线经过每条边刚好一次，请注意这两道题目的相似之处。）如此，可以将整张标准棋盘转化为图2-12所示的图，其中红色路线是欧拉的解答。

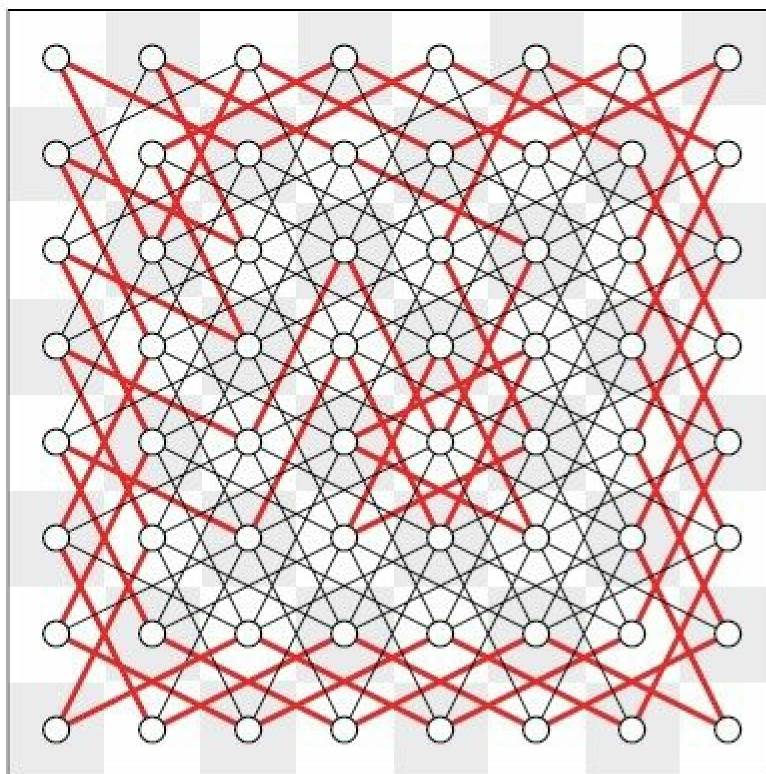


图**2-12**　用图表示的骑士周游路线

2.2.3 Icosian图

威廉·卢云·哈密顿爵士（Sir Wiliam Rowan Hamilton）是一位爱尔兰数学家。在欧拉之后一个世纪，哈密顿同样对某道与图相关的路线问题产生了兴趣。他研究了经过正十二面体全部20个顶点的路线，把正十二面体抽象为图2-13所示的图形，称之为**Icosian**，意即“有20条边的图形”。图上的线段代表正二十面体的棱，而标有字母的小圆圈代表顶点。

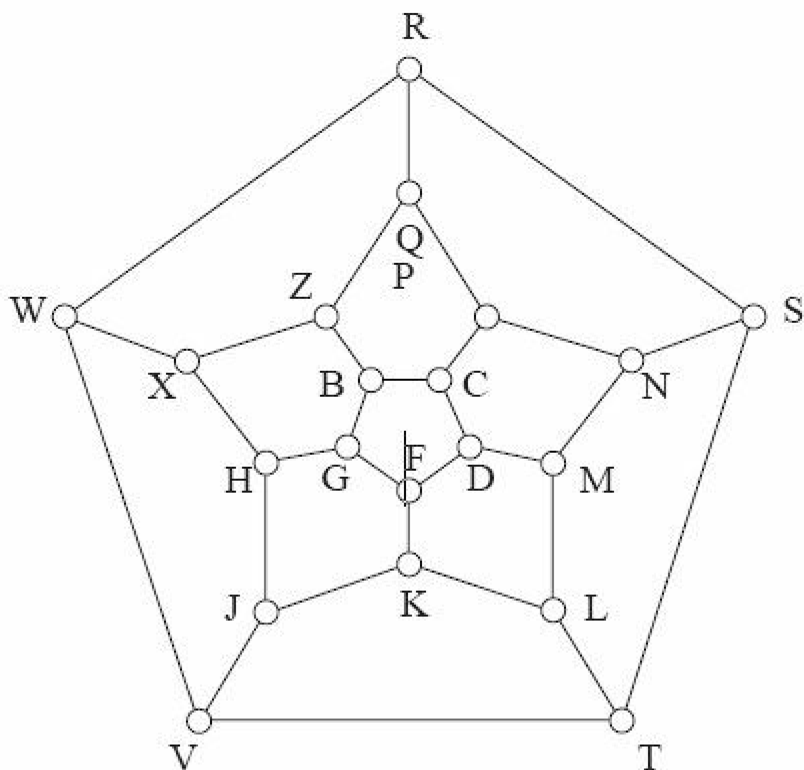


图**2-13　Icosian**图

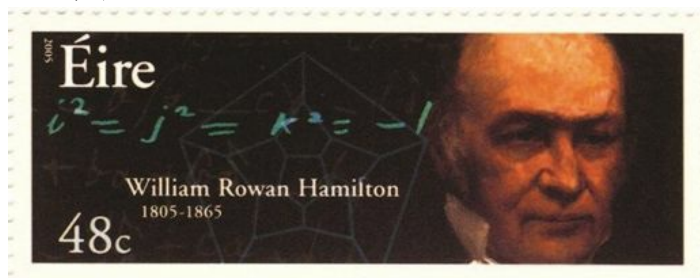


图**2-14**　哈密顿纪念邮票（爱尔兰邮局于**2005**年发行，哈密顿画像由爱尔兰皇家科学院提供）

哈密顿和欧拉一样，要求路线沿着Icosian图的边前进，依次经过各个顶点。但是哈密顿的做法非常有趣。他引入了一种代数系统来考虑图上的路线，与他对四元数的定义式在本质上高度一致。1856年，他在写给朋友John T. Graves的一封正式信函中描述了他的思路。[1]

1. Hamilton, W. R. 1856. In: H. Halberstam, R. E. Ingram, eds. *The Mathematical Papers of Sir William Rowan Hamilton*, Volume III. Cambridge University Press, Cambridge, UK. 612–624.

> 我将延续此前寄出的信里的做法，设三个符号ι、κ、λ满足下面的4个等式：
>
> $\iota^2=1$，$\kappa^3=1$，$\lambda^5=1$，$\lambda=\iota\kappa$

> 首先，我要举一两个例子来证明，这些符号在以上定义下，具有奇妙而确定的性质，从而足以成为合理的计算工具。对于它们得出的符号化的计算结果，我做了大量检验。在我看来，每一个结果都可以解读为古典几何学的某个多面体中连接各面或者各顶点的一条路线。这种理解方式很简单，而且往往很有趣。

这3个符号对应Icosian里的3种操作。符号之间做乘法相当于依次进行各个操作。[2]哈密顿利用这些符号进行计算，表明了无论从五个顶点中的哪一个出发，都有可能完成一条周游Icosian的路线。哈密顿非常喜欢Icosian图的结构，在信的末尾还向Graves介绍了一个在Icosian图上玩的游戏。

2. ι对应的操作就是在一条边上反转路线，κ对应在顶点处顺时针旋转到下一条边，λ对应在顶点处左转。因此，以连续做5次λ操作为例，实际就是在图中走出一条五边形路线，最后将回到原点，这就解释了为什么哈密顿定义$\lambda^5=1$。

> 我发现有些年轻人很喜欢玩一个新的数学游戏。这个游戏来自Icosian图，一个玩家选取任意连续5个点，比如abcde或者abcde'，插上5枚大头针，然后另一个玩家按照一条连续环状路线，再依次插入15根大头针，目的是到达剩下的每一个点，而且最后一枚大头针应当与对手插入的第一枚相邻。按照我在这封信里叙述的理论，符合条件的路线总是存在的。

后来，英国的一个玩具经销商推出了基于这个游戏的两款产品。第一款叫做“环游世界”游戏（Icosian Game），由一块木制棋盘和一些用来标注到过哪些点的象牙棋子组成。第二款则是一个手持玩具，名为“旅行者的十二面体——环游世界之旅”（Traveller's Dodecahedron: A Voyage Round the World），形状为压扁的十二面体，配有一些用来标记点的棋子，还有一条细绳用来勾勒路线。[3]



图**2-15**　**“**环游世界**”**游戏（左）和**“**旅行者的十二面体**”**游戏（右）（© 2009 Hordern-Dalgety Collection, puzzlemuseum.com）

3. 哈密顿游戏的两幅插图（图2-15和图2-16）由James Dalgety提供。他创办的智力玩具博物馆（Puzzle Museum）收集了各年代的智力玩具和游戏，藏品引人入胜。

虽然哈密顿热情高涨，他的游戏在市场上却一败涂地。你若亲自玩上几个回合，很快就会发现问题出在哪儿——Icosian图的路线实在太好找了。哈密顿极力为自己的游戏辩护，说他本人觉得玩起来相当有难度。看来，同样一个游戏，对小孩子来说很容易，对爱尔兰最伟大的数学家来说反而很难。之所以会出现这种奇怪的状况，可能是因为哈密顿思考时总是从代数的观点出发。说不定他玩游戏的时候，并不是用眼睛寻找路线，而是在脑海中进行ι、κ、λ操作来算出结果。

不过可喜的是，到了20世纪，哈密顿的游戏改头换面，成功地打开了销路。1975年，James Dalgety推出了“蠹虫之困”游戏（Worried Woodworm）。玩法也是要求玩家在一个图里找出路线，但是这次的图比较特别，路线很难发现。图2-16就是游戏使用的木制地图板。玩家的主要目标是以左下角的小孔为起点，右上角的小孔为终点，找到一条遍历板上每个小孔的路线。



图**2-16　“**蠹虫之困**”**游戏（**Worried Woodworm**）

（© 2009 Hordern-Dalgety Collection, puzzlemuseum.com）

Dalgety还补充了“蠹虫之困”游戏的其他挑战题。最近在解决这些难题的过程中，Concorde代码也派上了用场。如今的“玩家”在寻找周游23个点的蠹虫路线时，可以借助先进的TSP求解程序和高效的计算机。毫无疑问，光明磊落的玩家心里一定对这种做法不以为然。

2.2.4 哈密顿回路

不论是在欧拉的棋盘上周游的骑士，还是玩哈密顿的游戏的小孩子，他们都在寻找图中的路线。但是对于一般情形的问题，结果又如何呢？并不是所有的图里都能找到一条周游顶点的路线。判断哪些图有这样的路线，哪些图没有，则是一道难题。当年在数学领域中，图论才刚

刚开始寻求一席之地，哈密顿的鼎鼎大名让这道难题增色不少。正因为这段往事，他的名字在本节第一个登场。但是，不要因为我们冷落了欧拉而吓一跳。图论研究者暂时不用欧拉的名字，是为了把它留给另一类闭合路线，比如哥尼斯堡居民心驰神往的散步路线。因此，在一个图中，哈密顿回路（Hamiltonian circuit）就是经过所有顶点恰好一次的闭合路线，而欧拉回路（Eulerian walk）则是经过所有边恰好一次的闭合路线。[1]这两种回路都是图论中的基本概念，虽然表面很相似，实际意义却天差地别。

1. Eulerian walk是欧拉通路，欧拉回路应该是Eulerian circuit或者Eulerian cycle。本书原文没有明确区分通路和回路的概念，但是提醒读者注意。严格说来，回路是起点与终点相同的通路，通路本身未必是闭合路线。——译者注

要想判断一个图中是否存在哈密顿回路，是一个NP完全问题，复杂度与一般形式的TSP相当。然而对于欧拉回路的存在性，却存在一条简单的判定规则：图里可以有孤立顶点，不连接任何边，但除此之外，其他部分必须是连通的（connected），也就是说必须是连成一整片的，每个顶点都应该与偶数条边关联。

所以我们可以理解欧拉，却理解不了哈密顿。而且，年复一年，许多无畏的数学家提出了哈密顿回路的充分条件，可是他们的猜想最后都被推翻了。其中有一个例子非常著名，这就是P. G. Tait于19世纪80年代提出的猜想。当时，Alfred Kempe宣布证明了四色定理，得出结论说，至多只需要使用四种颜色，就可以给任意一张地图上色，使得任意两个有公共边界的相邻区域（国家或地区）都染成不同的颜色。这一发现令世人大为兴奋。Tait也受到了感染，并试图寻找另一种方式证明四种颜色就足够了。于是，他提出了一个猜想，认为在某一类图中一定存在哈密顿回路。

旅行路线和地图上色是如何联系起来的呢？把地图上每个区域的边界看作图的边，边界的交点看作图的顶点，你就明白了。生成的边界图里若存在一条哈密顿回路，则可以根据它来给地图染色。做法如图2-17所示，其中红色的边连成一条哈密顿回路。哈密顿回路不会自交，因此有内部和外部之分。位于回路内部的每条边都横穿内部区域，将其一分为二，从而可以只用两种颜色交替给内部区域染色，每经过一条边就换成另一种颜色即可；同理，回路外部的区域也可以只用另外两种颜色完成染色，最终就得到一幅四色地图。在图2-17中，内部区域使用深黄色和浅黄色染色，外部区域则用深蓝色和浅蓝色交替染色。

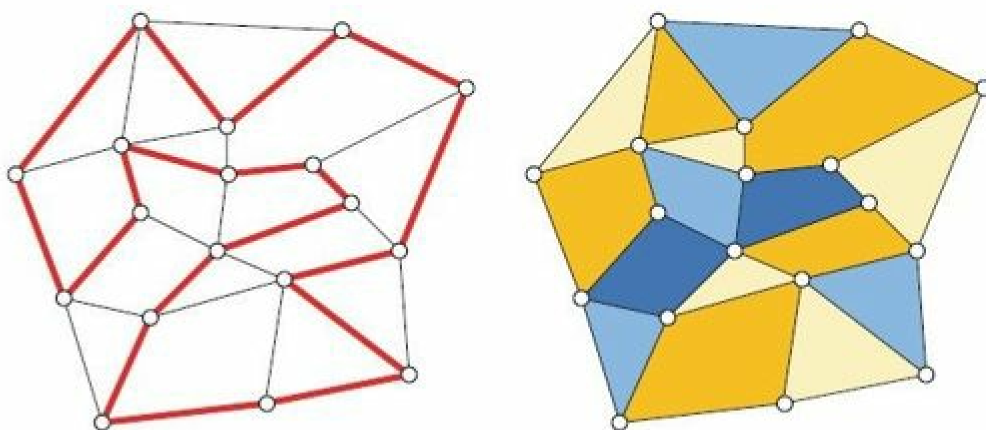


图**2-17**　通过哈密顿回路给地图染色

Tait知道，并非所有地图都存在沿着边界的哈密顿回路（美国本土地图就是一个现成的反例），但是可以通过一定的技巧，将四色问题的地图限定为边界图中每个顶点恰好连接3条边的情形，即3正则的（three-regular）。还可以进一步假定边界图是3连通的（three-connected），也就是说通过去掉一个或者两个顶点，可以把图分成两部分。Tait猜想，在这样的3正则、3连通的地图中，总是存在哈密顿回路。

William Tutte是伟大的图论专家，也是伟大的“布莱切利园”[2]密码破译专家。1946年，Tutte最终证明了Tait的猜想是错误的。这实在是非常遗憾，但是Kempe的四色证明早在1890年就被P. J. Heawood推翻了，相比之下，至少Tait的回路猜想坚持的时间更久。

2. “布莱切利园”（Bletchley Park）是二战期间英国政府情报部门进行密码破译的中心，现为英国国家计算博物馆所在地。——译者注

我要补充一点历史信息。四色问题的表述最早记载于哈密顿收到的一封信里。这封信的作者是Augustus De Morgan，时间是1852年。[3]哈密顿当时对这道题并没什么兴趣。他在回信里写道：“我最近不太可能尝试你的‘四元色问题’[4]。”

3. 哈密顿写给德·摩根的这封回信上的日期是1852年10月26日，在一本非常精彩的哈密顿传记作品中有记载，参见Thomas L. Hankins: *Sir William Rowan Hamilton*, Johns Hopkins University Press, 1980。

4. 此处哈密顿的表述为“quaternion of colours”，而quaternion表示哈密顿发明的四元数，故称“四元色问题”。——译者注

2.2.5 数学谱系

数学家都喜欢对学术传承关系追本溯源，从自己的博士生导师，到

导师当年的博士生导师，再继续这样一步一步追溯回去。数学谱系计划网站（The Mathematics Genealogy Project Web）由北达科他州立大学管理，收录了130 000多条博士生导师的记录，目标是汇集世界上所有数学家的信息。[1]我很自豪，因为我自己的学术源头可以追溯到维多利亚时代的数学家Arthur Cayley，然后可以不太正式地直接跳到哈密顿爵士本人。

1. 该网站地址为http://genealogy.math.ndsu.nodak.edu，截至2012年12月，已收录166 000多条记录。——译者注

我和Cayley之间的学术传承关系是直系的，回溯路径是：W. Cook（作者本人）——U.S.R. Murty——C. R. Rao——Ronald Fisher——James Jeans——Edmund Whittaker——Andrew Forsyth——Arthur Cayley。正式的追溯路径至此就结束了，因为Cayley当年学的是法律，并没有取得博士学位。但是他对数学深深着迷，并在1848年前往都柏林的圣三一学院，参加过哈密顿关于四元数的讲座。他受到哈密顿的影响，一面从事法律工作，一面继续数学研究，总共完成了几百篇数学论文，因此于1863年被任命为剑桥大学的赛德勒数学教授[2]。尽管Cayley并没有继承哈密顿对TSP相关问题的兴趣，但是他却是图论领域的著名人物。我们稍后会讲的“树”的概念就是由他提出的。

2. 赛德勒数学教授（Sadleirian chair of mathematics）是剑桥大学纯数学教授荣誉职位，因Lady Sadleir遗愿而设立并得名，Cayley是第一位获此殊荣的数学家。——译者注

2.3 维也纳—哈佛—普林斯顿

欧拉和哈密顿都曾研究路线，但是从国际象棋棋盘和正十二面体到上路奔波的旅行商之间却有天壤之别。对旅行商来说，可不是随便哪条路线都能满意。他们只想要总路程最短的那一条。

为了讨论实际旅行成本，我们要前往奥地利的维也纳，拜访Karl Menger，了解他的研究工作。在20世纪20年代，Menger最关心的研究题目包括一项：度量空间曲线长度的方法。 可能是在这项内容艰深的研究的启发下，1930年2月5日举办的一场学术讨论会上，Menger公布了一道与TSP密切相关的题目[1]。

1. Menger (1931).

> 我们用“邮递员问题”（messenger problem）一词表示这样一个问题（因为每位邮递员在实际工作中都会遇到这个问题，不过许多旅行者也会遇到）：给定有限数目的点，它们两两之间的距离已知，试求连接所有点的最短路线。

这道题也就是要找到一条周游所有点的路线，但不需要最终回到起点。所以只需要在终点和起点之间添加一个“虚城市”，就可以把通路两端连接起来，形成一条回路，从而转化为TSP路线。可以设“虚城市”和任意一座真实的城市之间的路费都为零，这样一来，虽然增加了一座城市，但并不会干扰我们选择路线的起点和终点。

在维也纳数学研讨会的出版文献资料中，留下了关于Menger邮递员问题的德语文献记录。这一问题的发表显然具有重要的历史地位，但是美国的TSP研究热潮似乎并不直接来源于此，而应该归功于哈佛大学的优秀数学家Hassler Whitney。Dantzig、Fulkerson和Johnson的经典论文里提到了Whitney的讲座。[2]

2. Dantzig, Fulkerson, Johnson (1954).

> 哥伦比亚大学的Merril Flood多次介绍旅行商问题，在激发相关研究兴趣方面功不可没。早在1937年，他便努力求解校车路线的近优解。Flood和普林斯顿大学的Albert. W. Tucker都回忆说，他们最早得知这道题是在1934年，当时Hassler Whitney在普林斯顿大学做了一次学术报告（尽管近期向Whitney本人求证时，他似乎对此全无印象）。

Merril Flood在叙述TSP的研究历史时，也把功劳归于Whitney的讲

座。在1956年的一篇论文里，他写道："该问题于1934年由Hassler Whitney在普林斯顿大学的一场学术报告上提出。"[3]很久之后，到了1984年，Flood在接受Tucker的访谈时，仍然用"Whitney的48州问题"代指TSP。[4]

3. Flood, M. M. 1956. Operations Research 4, 61–75.

4. Flood, M. M. 1984. The Princeton Mathematics Community in the 1930s, Transcript Number 11 (PMC11). Princeton University.

人们会很自然地猜想，在Menger的维也纳研讨会和Whitney的普林斯顿报告会之间，可能存在某种联系。Alexander Schrijver发现了证据。他指出，在1930年到1931学年间，Menger曾到Whitney所在的哈佛大学访问了一学期[5]，因此他们两人曾经见面。然而，至于他们在交流中是否直接提到了旅行商问题（邮递员问题），目前仍不得而知。

5. Schrijver（2003）一文中详细地记录了Menger的具体工作。

同样，Whitney当年在普林斯顿大学是否真的讨论过TSP，也依然没有定论。很遗憾，普林斯顿大学并未保留20世纪30年代的数学系学术研讨会记录。不过，哈佛大学普西图书馆却有另外一些档案。这里收藏了Whitney的论文，文档总体积达到0.1立方米。其中有一批手写的笔记写于1930年后不久，似乎是Whitney为学术研讨会做准备时写的。笔记中介绍了图论，摘录一段如下。

> 下面是一个难度更大的类似问题：我们能否在一个图中找出一条简单闭合曲线，要求它恰好经过每个顶点一次？该问题对应另一个问题，即，给定一组国家，是否可能以某种方式周游列国，从而恰好到达每个国家一次？

把每个国家看作一个顶点，若两个国家之间有公共边界，就在对应的两个顶点之间连一条边，便得到了第一个问题中的图。第二个问题中周游列国的路线就是图里的一条哈密顿回路。用这样的例子来描述哈密顿回路问题，其实并不是无心之选，而且这道题距离真正的TSP显然只有一步之遥。

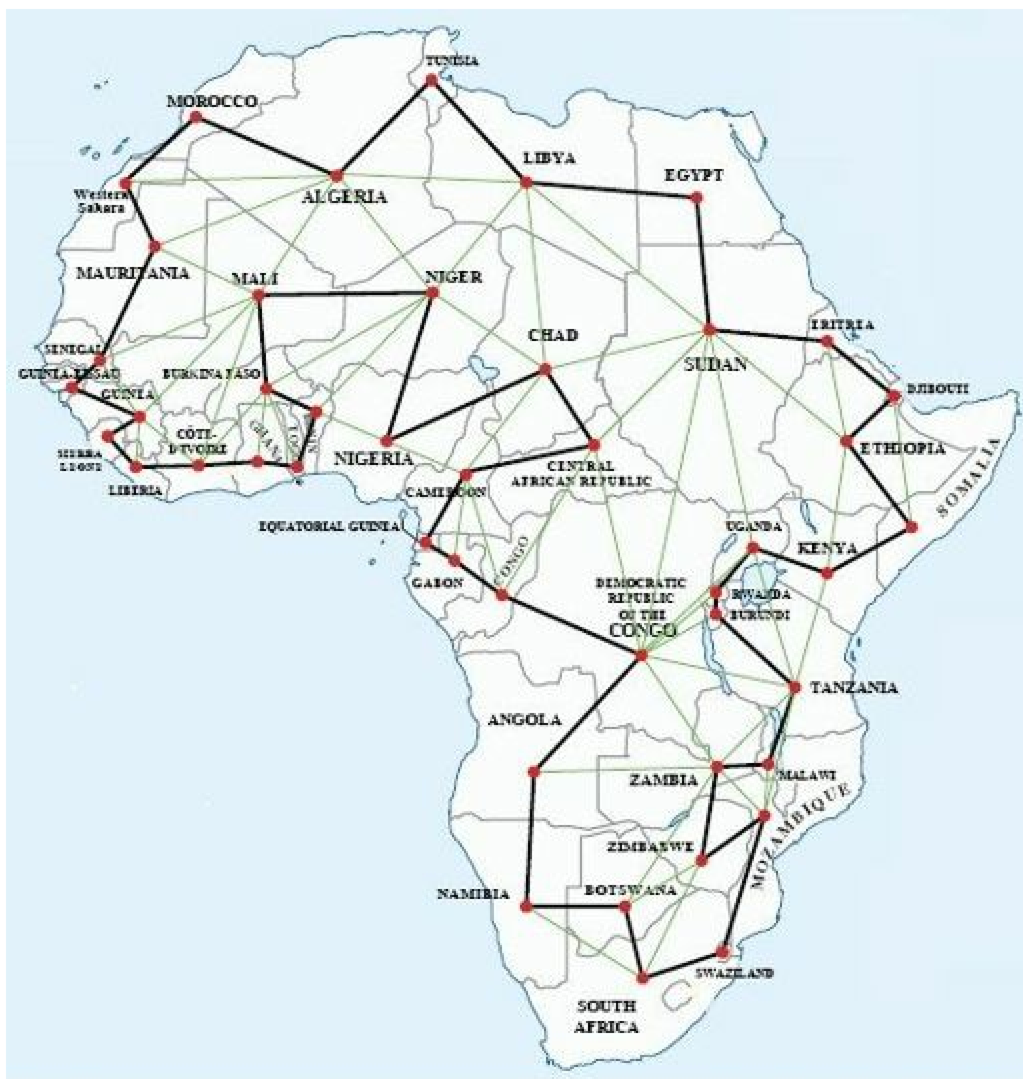


图**2-18**　周游非洲的哈密顿回路

惠特尼例题具有地理背景，与Flood回忆起的环游美国48州问题彼此吻合。确实，惠特尼用来说明哈密顿回路的例子很可能就是美国TSP研究的源头。正如Alan Hoffman和Philip Wolfe评价的那样，惠特尼把旅行商介绍给数学界的过程中，角色“也许相当于门格派来的邮递员”[6]。

6. Hoffman and Wolfe (1985).

2.4 兰德公司

在20世纪30年代晚期和40年代初期，相关问题的研究记录中并没有出现“旅行商问题”或“TSP”的字眼。但是到了40年代末，这道难题却已声名远扬。此时，求解TSP的作战中心已经从普林斯顿大学转移到了兰德公司，Flood也同样搬到了加利福尼亚州。

2008年12月，普林斯顿大学的Harold Kuhn在一封电子邮件里如此写道：

> 在1949年的数学系范氏大楼[1]里，旅行商问题的名号已经颇为响亮。举个例子，兰德公司当时现金悬赏求解一些问题，它就是其中之一。我记得他们那张列表贴在范氏大楼里的公告板上，那是1948至1949学年的事了。

1. 范氏大楼（Fine Hall）是普林斯顿大学数学系所在地，因此也代指该校数学系。该楼得名于首任系主任Henry Burchard Fine，他是经典著作《范氏大代数》（*A College Algebra*）的作者。——译者注

兰德公司的奖金悬赏列表！有关TSP的文献资料里经常提到这些奖，但是现在已经很难找到当年那份文件的副本了。Hoffman和Wolfe称，兰德公司设立奖项是“为了TSP的理论研究”。尽管从未有人获得奖金，但是那张奖金列表再加上兰德公司研究小组的极高声望，确实对于宣传推广TSP作出了重要的贡献。

在兰德公司内部员工的口中，还出现了另一个奖项。著名女数学家Julia Robinson评价自己在博弈论方面的研究工作时提到了它：“兰德公司悬赏200美元求解那道题。在我的论文“迭代法求解一个游戏”（An iterative method of solving a game）里，我证明了我的方法确实是收敛的，但是他们也没给我奖金，因为我是兰德公司的内部员工。”[2]

2. Reid (1996).

Robinson很可能是受到了列表上另一个问题的启发，于是在1949年开始研究TSP。那段时间，她在一份手写笔记里描述了一个通用的数学方法。她对旅行商问题的研究工作与这个通用方法是一致的。“有些问题的叙述相当简单，但是大家完全不知道用什么样的方法可以求出结果，我就更喜欢钻研这样的问题，而不太喜欢只需要推广现有解法的问题。”[3]当然，TSP正合她的心意，因为在Menger的研讨会之后接近20年的时间里，一直都没有人发表这方面的研究进展。而Robinson后来作出了不少贡献，兰德公司几年后正是在她的基础上取得了突破。我们将在

第5章介绍具体内容。

3. Reid (1996).

TSP从何得名

1949年，Robinson在一篇论文里很自然地使用了“旅行商问题”的说法，可见这个概念在当时已经广为人知。事实上，在现有的TSP文献中，她的这篇论文是使用这一说法的最早的一篇，只有兰德公司悬赏列表的副本才能动摇这一历史地位，但是得先有人从故纸堆中翻出那份列表来才行。Robinson将这道题表述为寻找“旅行商以华盛顿为起点和终点、周游各州首府的最短的路线”，与Flood的回忆相吻合，与Dantzig等人使用的数据集也是一致的。

Robinson叙述这个问题的方式把TSP和环游美国48州问题联系在了一起，但是我们依然不知道“旅行商”到底是何时出场的，“旅行商问题”的名字最早是何时开始使用的。Merrill Flood似乎应该知道答案，然而很不幸，他却对此一无所知。他对Albert Tucker解释说：“我不知道是谁把惠特尼的问题称为旅行商问题，是谁创造出这么生动有趣的名字。但是人们显然都很认可这个名字，后来也发现这个问题其实非常基本、非常重要。”无论旅行商问题的名字是如何产生的，到了20世纪50年代中期，这个名字已经得到了广泛使用，只不过“traveling salesman problem”的英文拼写有时存在细微不同，比如写成“travelling”或者“salesman's”之类的[1]。同时，这个问题的“恶名”也逐渐远扬。万事俱备，只等Dantzig等人登场。

1. 同样，这道题的中文译名也并不唯一。除了本书使用的“旅行商问题”外，常见译名还包括“旅行推销员问题”和颇有趣味的“货郎担问题”。——译者注

2.5 统计学观点

在数学界，常常有各路人马研究同一个重要问题，有时研究小组之间并无交流，都以为自己是在孤军奋战。旅行商问题也不例外。当时在美国，Flood和兰德公司都在奋力求解TSP。与此同时，在地球的另一端，统计学家Prasanta Chandra Mahalanobis也开始研究这个问题。但Mahalanobis的数学观点别具匠心，关心的实际应用问题也与美国研究者截然不同。

2.5.1 孟加拉黄麻农田

Mahalanobis成立了印度中央统计局，创办了统计学期刊*Sankhya*，被誉为印度的统计学之父。他的一个主要研究方向是抽样调查的技术方法，并在此领域与TSP结下了不解之缘。

20世纪30年代，黄麻作物是印度政府财政收入的重要来源，大约占到出口总额的四分之一。当时，黄麻种植地主要分布在孟加拉地区[1]。如何收集数据以准确预测黄麻产量是实际生产中的一大问题。

1. 当时孟加拉地区遭受英国殖民统治，是英属印度的一个省。——译者注

由于当时黄麻种植分散，孟加拉地区总共有大约600万处小农田生产黄麻，因此要想完全调查每一处黄麻田是不现实的。Mahalanobis提出了一个替代方案：随机抽样调查，把孟加拉地区分成多个区域，各个区域内的田地性质都差不多，在每个区域随机选一些点观察黄麻的种植情况即可。抽样调查需要花费时间和金钱，成本的主要影响因素就是在不同样本区域之间运送人和设备的时间。从这个方面来看，这个问题就变成了一道TSP，要寻找连接所有选定农田地点的高效路线。就此，Mahalanobis在1940年的一篇研究报告里写了下面这段话[2]：

2. Mahalanobis (1940).

> 很容易大致了解旅行的情况。首先假设在任意地区内都随机分布着n个样本单位，每个样本单位都可以看成一个几何学上的点[3]。再假定在两个样本点之间移动时，可以安排路线使总路程尽可能小，换言之，在任意两个样本点之间都沿直线移动。如此一来，容易知道，任意两个样本点之间的路程的数学期望是$\sqrt{n}-1/\sqrt{n}$，所以我们从一处样本田去往另一处的花费就正比于$\sqrt{n}-1/\sqrt{n}$。如果n比较大，也就是说，我们考虑的地区面积相当大，那么从一处样本田去往另一处所需的时

间将近似正比于$\sqrt{n}$，这里n是给定地区内的样本总数。

3. 在几何学中，点只有位置，没有大小和形状。——译者注

什么是期望？如果我们每次都随机取n个点并求解对应的TSP题目，那么经过多次重复试验，最优路线的平均长度就是期望。也许因为Mahalanobis是一位统计学家，他的研究兴趣在于统计学领域，所以他并没有讨论实际操作任务，换言之，没有对于具体数据真正求出路线，而是关注最优路线长度的统计估计结果。与普林斯顿大学和兰德公司的研究方向相比，这种视角可谓独树一帜。

在计算孟加拉地区抽样调查的预计成本时，Mahalanobis提出的估计值派上了用场。1937年，人们进行了一次小型试验，并在次年实施了大规模的调查。在做出这些决定的过程中，预计成本都是非常重要的考虑因素。

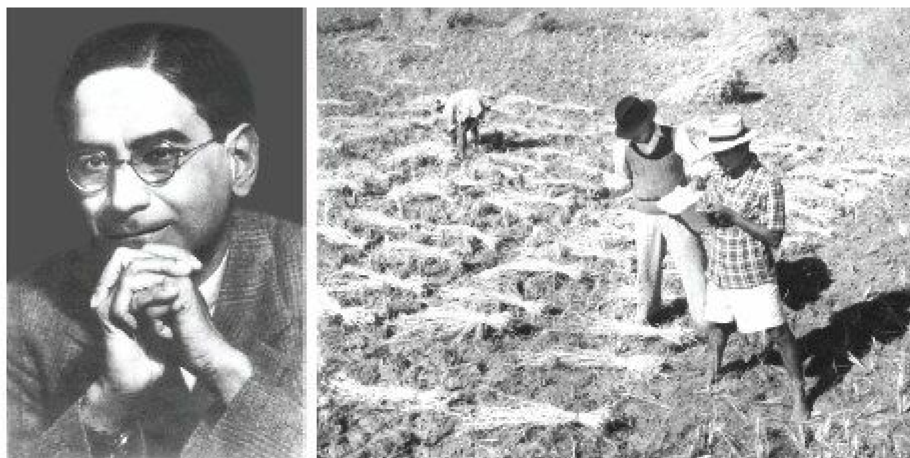


图**2-19　Prasanta Chandra Mahalanobis**（左）和**Mahalanobis**在进行田间抽样调查（右）。（印度中央统计局**Mahalanobis**博物馆供图）

2.5.2 证实路线估计值

Mahalanobis给出了一个TSP公式，却并没有给出严谨明确的分析过程。不过，他的研究工作给统计学界的后来人指明了前进的方向，确立了研究的目标。每一个点的坐标设为(x, y)，其中x和y分别等可能地取0到1之间的任意数值，像这样的分布称为随机分布。统计学家想要进一步了解的问题就是，当单位正方形内随机分布的城市数目增多时，TSP路线会如何增长。具体说来，他们想求出周游这个点集的最优路线长度满足什么公式。

研究者在两个方向分别取得了进展。1948年，Eli Marks证明了周游一组随机点的最优路线长度的期望不小于$(\sqrt{n}-1/\sqrt{n})/\sqrt{2}$；1949年，M. N. Ghosh证明了期望长度至多不超过$1.27\sqrt{n}$。当n很大时，综合以上两个结果便证实了Mahalanobis的直觉，路线长度的期望值确确实实正比于$\sqrt{n}$。

Ghosh在论文里给出了期望的上界，并且特意对根据具体数据实际计算结果的工作发表了评论。“在地图中选定该区域的n个随机分布的点以后，很难真正实际找出连接所有点的最短路线，除非n非常小。而在大规模调查中，几乎不可能出现很小的n。”[1]

1. Ghosh, M. N. 1949. Calcutta Stat. Assoc. 2, 83–87.

有趣的是，Ghosh也发现了TSP的真正困难之处在于找出最优路线，但是他显然没有受到Menger、Whitney和Flood的影响，而是独立得出了这个结论。

2.5.3 TSP常数

综合Mahalanobis、Marks、Ghosh三人的结论，我们可以对平均路线长度做出估计，但是无法从估计值中看出一系列试验所得长度的分布范围，因为有些试验的随机点集可能给出很长的最优路线，有些则可能很短。不过如果n充分大，那么这种偏差很大的现象并不会出现。图2-20的柱状图可以帮助理解这一结论。对1000个点的情形，进行10 000次随机取点试验，把每次的最优路线长度除以$\sqrt{1000}$就得到了图中纵坐标的数值。所有试验值形成了一条优美的钟形曲线，正中间的均值是0.7313。在这幅图中只有1000座城市，所以有些路线长度值会有一定程度的偏离。但是根据Beardwood、Halton、Hammersley于1959年提出的著名理论，可以得知，如果n增大，则长度数值分布会以特定的β为中心形成狭窄的峰形[1]，β称为**TSP**常数（TSP constant）。

1. 结论为，如果n增大，则最优路线长度除以$\sqrt{n}$的值会以概率1收敛到β。Beardwood, J., J. H. Halton, J. M. Hammersley. 1959. P. Camb. Philos. Soc. 55, 299–327.

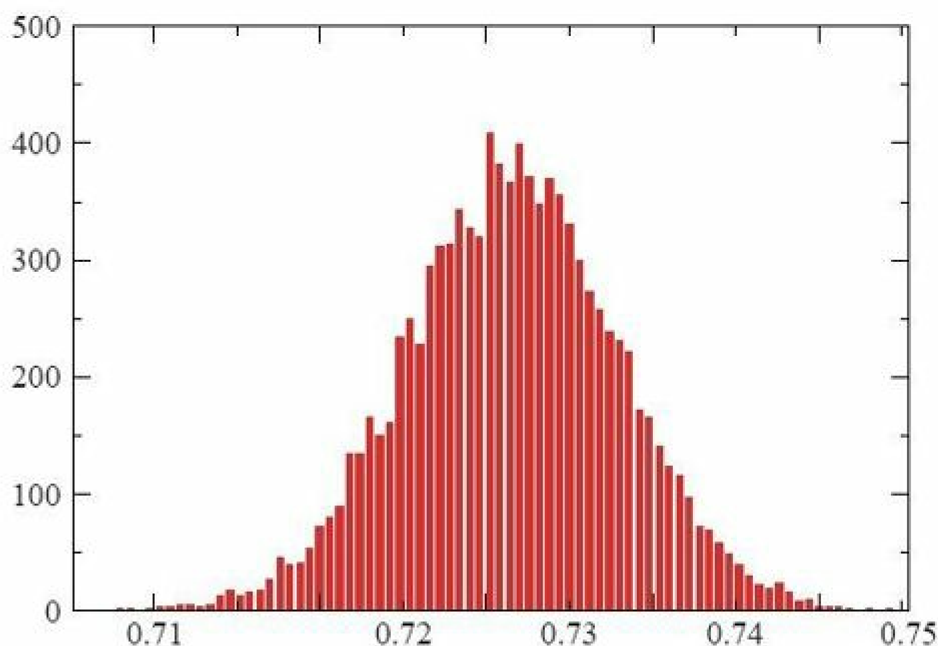


图**2-20**　对于**1000**个点的情形进行**10 000**次试验的路线长度分布图

确定TSP常数β的值是一个很有吸引力的挑战，这方面的研究也为概率论带来了重要的新分支学科。但是现有的结果都只是近似值，要想确定真实值依然有很远的路要走。所以现状是，我们知道一个自然常数，却不知道它的真实值。

David Applegate、David Johnson、Neil Sloane正在研究β。他们的工作已经解决了6亿多道几何情形的TSP题目。解题过程中，Concorde也得到了充分的锻炼。尽管只凭一大堆计算结果并不足以证明任何确定的结论，但是得到的数据曲线（例如图2-21）都强烈表明，当n增大时，平均路线长度除以$\sqrt{n}$的数值会持续下降，最终趋于固定值β，近似为0.712。[2]

2. 物理学家也对β进行了耐人寻味的研究，参见：Percus, A. G., O. C. Martin. 1996. Phys. Rev. Let. 76, 1188–1191.

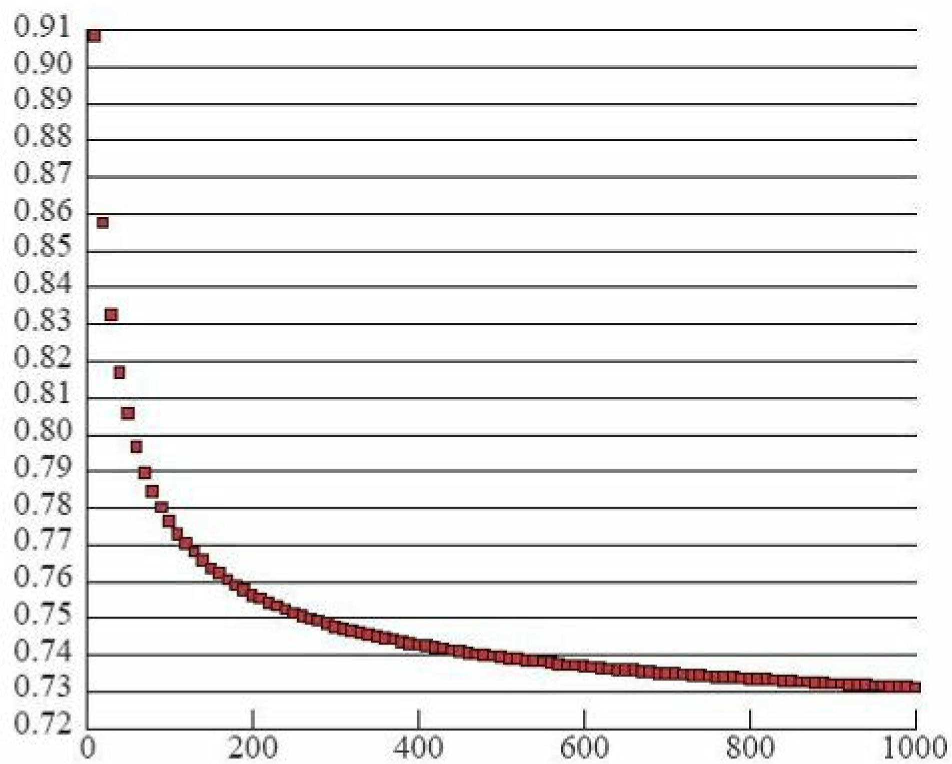


图**2-21**　平均路线长度与***n***的关系图，其中每个点都为**10 000**次试验的平均值

第3章 旅行商的用武之地

> 我的数学题目最初起源于实际问题，但它对我的吸引力并没有因此削弱，而是恰好相反。

——George Dantzig，1986年[1]

1. Albers, Reid (1986).

旅行商问题的实用本质从其名称就体现得淋漓尽致。这一特点无疑也促使人们关注该问题的计算方面，从而完全避开了约翰·冯·诺依曼所谓的危机："换言之，数学题目若远离其实践根源，或历经大量孤立的理论思考，则有退化变质之虞。"这句话非常著名，出自他的文章"The Mathematician"。事实上，实际应用问题层出不穷，不断为TSP研究领域注入新鲜的生命力，一直推动TSP研究滚滚向前。

3.1 公路旅行

本章将综述TSP的应用，首先介绍一些实际的旅行事例，当然也包括旅行商的旅行。

3.1.1 数字化时代的推销员

如今，各地区的巡回推销员出行时，一般会驾驶配有GPS（全球定位系统）的汽车。在车载GPS使用的地图软件中，常常包含一个小程序，能够求解十几个城市的小规模TSP。这个数目往往已经足够满足单日出行的需要了。GPS设备里存有详细的地图，从而能够准确地估计从某地去往另一地所需的时间，因此TSP的解便可以反映出实际面临的行车条件。

普林斯顿大学的Alain Kornhauser是地图制图技术应用方面的专家。他介绍了GPS设备的一种有趣的反向用途。使用者在GPS上按照精度和纬度设定目的地之后，有时候会发现这个点不可能落在已知的道路区域内，也就是说根本没有路可以到达那里。但是如果当地的货车司机必须把货物送到，他往往能找到一条路成功抵达目的地，比如抄一条地图上没有标注的小路。这种情况下，GPS系统就把信息报告给中心服务器，然后服务器就在相应的地图坐标格里，沿着货车经过的路径，加上一条新的连接线。于是下一次有人要去同一个地点送货的时候，地图软件就可以用到这条新添加的路了。

3.1.2 取货与送货

人们常常用小规模的TSP模型来规划客车接送乘客或者货车配送货物的路线。Merrill Flood曾经写道，当初是一道校车路线问题引领他走上了TSP研究的道路。伦敦政治经济学院的George Morton和Ailsa Land组成了另一个早期的研究小组，他们对TSP的兴趣则来源于一道洗衣店送货的题目。Rapidis公司开始的时间更晚一些。该公司之所以使用Concorde程序，是为了给客户规划路线。他们的客户是广告材料样品分发商Forbruger-Kontakt公司，其业务范围包括丹麦等多个国家。Rapidis公司开发了一款路线规划软件，图3-1就是根据该软件的屏幕截图而画

成的。图中的路线需要遵守单行道之类的行车限制，因此在两地点间来回的花费与方向也有关系。

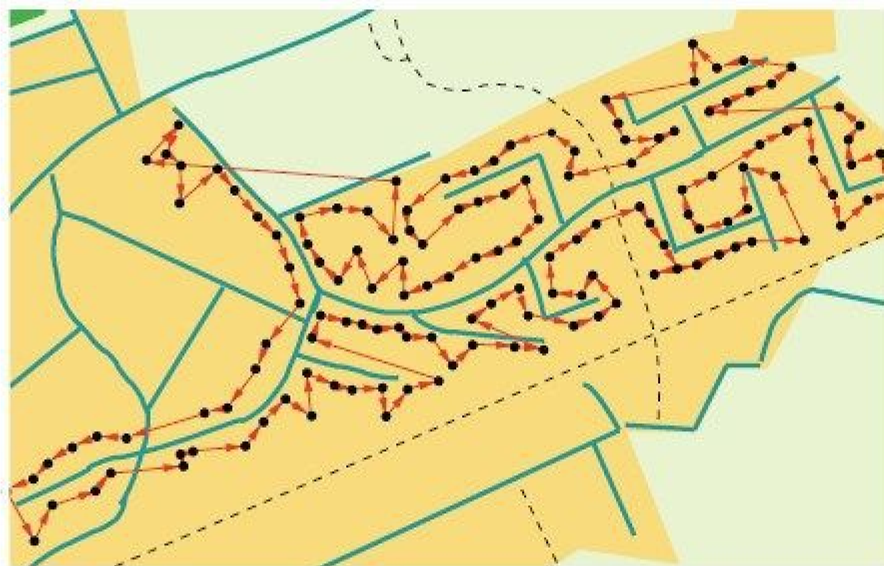


图**3-1　Forbruger-Kontakt**公司送货的**TSP**路线（**Thomas Isrealsen**供图）

3.1.3 送餐到家

佐治亚理工学院的一个研究小组介绍了TSP算法的成功应用案例。在美国亚特兰大市的“送餐到家”（Meals on Wheels）服务项目中，该研究组用一种快速启发式算法为送餐工作人员构建路线。[1]该项目每天要把饭送到大约200个地点，其中每一名司机配送的地点在30到40之间。为了构建司机的行车路线，研究者首先将所有200个地点连成一条路线，然后再把这条总路线分成长度合适的小段路线。借助如图3-2所示的空间填充曲线，可以得到总路线。如果让空间填充曲线的结构变得越来越精细，那么最终城市内任意一点都将落在这条曲线上。按照曲线经过各点的顺序，把200个不同地点排起来，就得到这条启发式路线。

1 “送餐到家”是一种社会服务项目，志愿者为老年人及行动不便者提供午餐并负责配送上门。参见：Bartholdi III, J. J., et al. 1983. Interfaces 13, No. 3, 1-8.

这种构造路线的方法操作简单，因此容易更新维护。如果有新的客户加入或者原来的客户离开，项目管理者可以轻而易举地手动更新路线，下面简单介绍一下他的做法。首先，一个地点在空间填充曲线上的相对位置θ完全决定了它在路线中的次序，所以佐治亚理工学院的研究小组就先对地图上密密麻麻的小格预先计算好θ值，这些小格也就是位

于亚特兰大标准地图不同坐标(x, y)处的地点。然后管理者把当前参与项目的客户名单储存在两套索引卡上，第一套按姓名首字母顺序排列，第二套则按照路线中所处的次序排列，也就是按照θ从小到大排列。这样一来，如果想要去掉一个客户，只需要抽出相应的两张卡片即可；如果想要新增加一个客户，则对照地图确定新客户位置对应的(x, y)坐标，然后在计算出来的θ值表格里查找相应的θ值，从而确定新客户在路线中的次序并把其卡片插入第二套卡片中。对于这道TSP实际应用题，他们的解法不需要借助高科技，实在是巧妙至极。

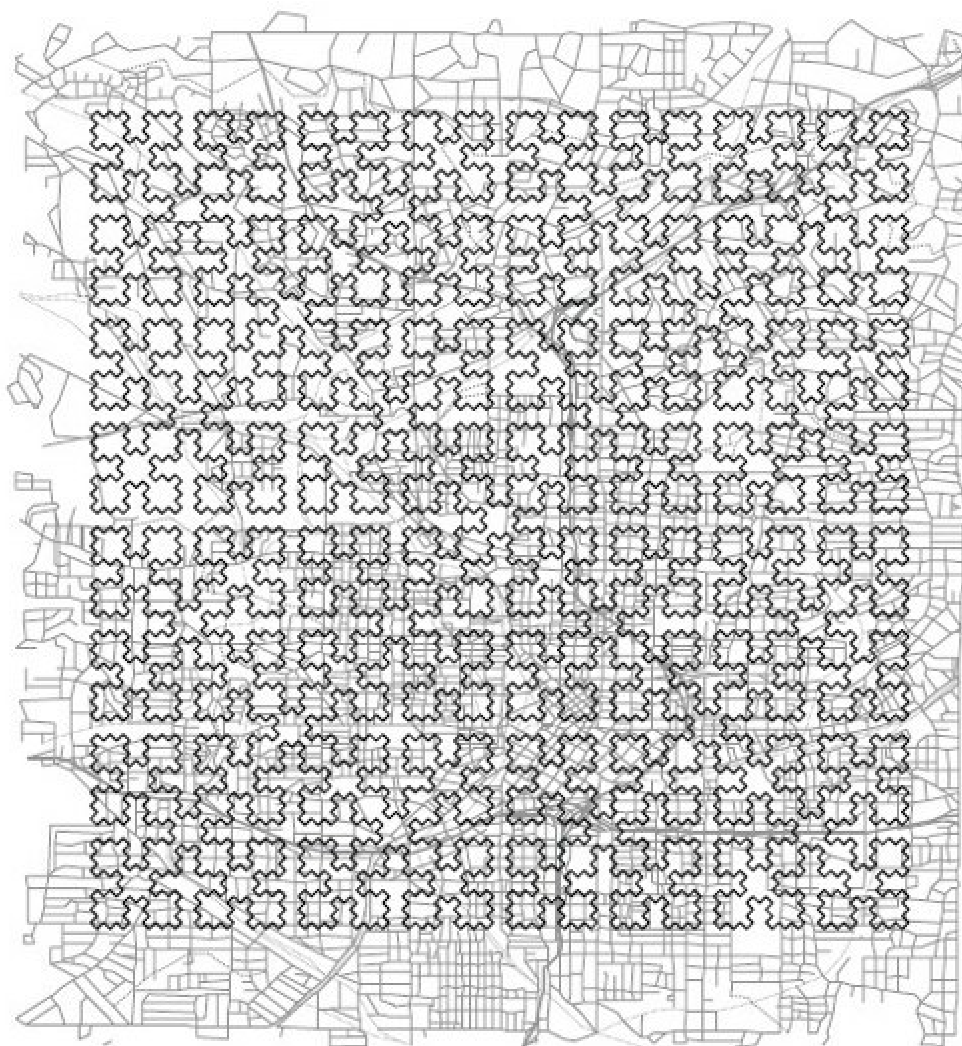


图**3-2**　亚特兰大地区的空间填充曲线（**John Bartholdi**供图）

3.1.4 农场、油田、蓝蟹

TSP可以用于规划另一类路线，就是检查、视察相距较远的地区。此类后勤组织应用以Mahalanobis在20世纪30年代的农业研究为早期实例，不过也可以用于其他情况。William Pulleyblank在论文中介绍了一例。题目背景是为油田规划路线，总共到达尼日利亚的47个海上采油平台，而乘坐的直升机要从陆上基地起飞，最后TSP软件解得了所求路线。马里兰大学的一个研究小组则提供了另一例。这道问题是关于乘船出海的。他们需要到达美国切萨皮克湾的大约200个监测站，以监测海湾内的蓝蟹数量，结果遇到了困难，由于每次出海时间太长，没法完成对所有检测站的高密度监测。因此他们转向TSP模型求助，解决了路线规划问题。

3.1.5 巡回售书

小说《毗湿奴之死》（*The Death of Vishnu*）的作者Manil Suri是一名数学教授。他在*SIAM News*发表了以下感想[1]。

1 Suri, M. 2001. SIAM News 34, p. 1.

> 我在美国的首轮巡回售书将从2001年1月24日开始，历时三周，总共要去13座城市。出版社给我城市列表的时候，我发现这简直是太神奇了！我自己要亲身体验旅行商问题了！我竭力把内心的激动之情传达给出版社的宣传部，努力向他们描述这一切在数学上有多么重要，我们要怎么做才可能得出一个最优解，如此等等。我的狂热弄得他们很不自在。他们告诉我，他们在规划巡回路线方面经验丰富，要我放心，还说如果需要数学上的帮助的话，他们会再和我联系。至今，他们也没有找我。

尽管Suri的出版商对此全无热情，但巡回售书作为TSP的背景问题确实非常自然。

3.1.6 “多走一里路”

虽说 “多走一里路”俱乐部（Extra Miler Club）的座右铭是“因为两点之间的最短距离太没劲”[1]，不过他们的会员以周游美国全部3100多个县（郡）为目标，而且确实喜欢规划路线。严格说来，这算不上TSP，因为只要进入县的边境线就算到过这个县，不过也有一些会员认为要到政

府所在地才真正算数。

1 该俱乐部位于美国，会员多数以驾驶机动车周游全国为目标。在英文中，“多走一里路”（extra mile）指超出别人期许，多付出一些努力，出自圣经马太福音：“有人强逼你走一里路，你就同他走二里。”——译者注

据《华尔街日报》报道，“多走一里路”俱乐部有一名会员的目标是到北美的每一家麦当劳吃一个巨无霸，总共有13 000多家店。[2]这本来是个不错的TSP实例，可惜俱乐部官方网站介绍说，这位会员“现在已经向着饕餮难度更低的目标出发了”。

2 Dosher, M. *Wall Street Journal*, 1998-02-09, B1.

3.1.7 摩托车拉力赛

“多走一里路”俱乐部的会员通常开汽车出行，而铁臀摩托车协会（Iron Butt Association）则流行选择摩托车作为交通工具。该协会实力雄厚，会员多达35 000人，挑战项目也不少，其中一项就是“十天四十八州骑行之旅”，又称为“48/10之旅”，也就是在10天时间内，环游美国本土的48个州。骑手可以选择任何路线，但是为了证明自己到过各州，必须拿到加油站收据之类的书面证据。2009年2月，一名女骑手Maura Gatensby发来一封电子邮件，想知道Dantzig-Fulkerson-Johnson的TSP路线都经过了哪些城市。

> 多数人研究这个问题的时候，都是拿到已有的地点和路线，然后想方设法削减路程。而对我来说，了解到数学中的这道题之后，我更愿意从Dantzig路线出发，当然也许会尝试移动某些地点来缩短距离，但是如果Dantzig路线并不算太长的话，我就干脆直接选定它，因为它具有重要的历史意义。有时候“最短的”路线未必就是“最好的”路线，因为追随大师的足迹更具有诗意和韵味。

如此将他们的最优解付诸实践，实在是恰到好处。

Gatensby女士在邮件中提到，迄今为止，“10天48州之旅”最短的路线是6967英里（11 212千米）。虽然对于今天的TSP求解程序来说，48个城市的情形并不难解决，但是这道题目还是很复杂，因为在各州都有很多城市可供选择。如果限定一些条件，比如各州目的地只限制在一系列已知的加油站范围内，那么寻找最优路线的挑战应该很有意思。

3.1.8 飞行时间

Ron Schreck的速度记录也许无人能敌。他住在北卡罗来纳州，驾驶RV-8飞机“Miss Izzy”四处飞行。2007年，他萌生了一个点子，决定在一整天内到达全州总共109家公共机场。Concorde程序为他提供了一条路线，他又稍作改动，使自己能在日出之前先到达几处设有跑道照明系统的机场。实际飞行的日子定在7月4日，因为那天是美国的公共假期，可以帮他避免延误起降时间。他当天的总飞行距离为1991英里（约3204公里），总时间为17个小时，平均每两次着陆之间只相隔9分半钟。这里所谓的“着陆”不是真正落地，基本上只是轮子接触地面一下，然后飞机立刻升回空中。



图**3-3**　驾驶**Miss Izzy**飞行途中（**Ron Schreck**供图）

3.2 绘制基因组图谱

除了行人车辆东奔西走的情境以外，我们还发现TSP模型具有某些相当不可思议的应用。其中，TSP在遗传学领域的应用最为引人注目。染色体上的标记是基因组图谱的地标，所以过去十年间，精确定位这些标记一直是研究的热点。

在基因组图谱中，每一条染色体上都按顺序排列了一些标记，相邻两个标记之间的距离也都有估测的数值。这些图谱中的标记实际上是DNA片段，恰好只在整个基因组里出现一次，而且可以通过实验室研究工作准确检测到。研究人员利用了它们的唯一性和可检测性，对各个实验室绘制的物理图谱进行核实、对照和合并。研究的重中之重是要精确了解这些标记在染色体上出现的顺序，TSP正是在这一步大显身手。

可以通过不同方法得到这些标记相对位置的实验室数据，其中有一种非常重要的技术称为辐射杂种细胞作图（radiation hybrid mapping），简称RH作图技术。这种方法分为两步。首先将染色体暴露在高剂量的X射线里，使之受到辐射后断裂成若干片段。然后把这些片段和啮齿目动物的遗传物质组合在一起，形成杂交细胞系，再分析确定标记位置。图3-4是这两步操作的简单示意图。

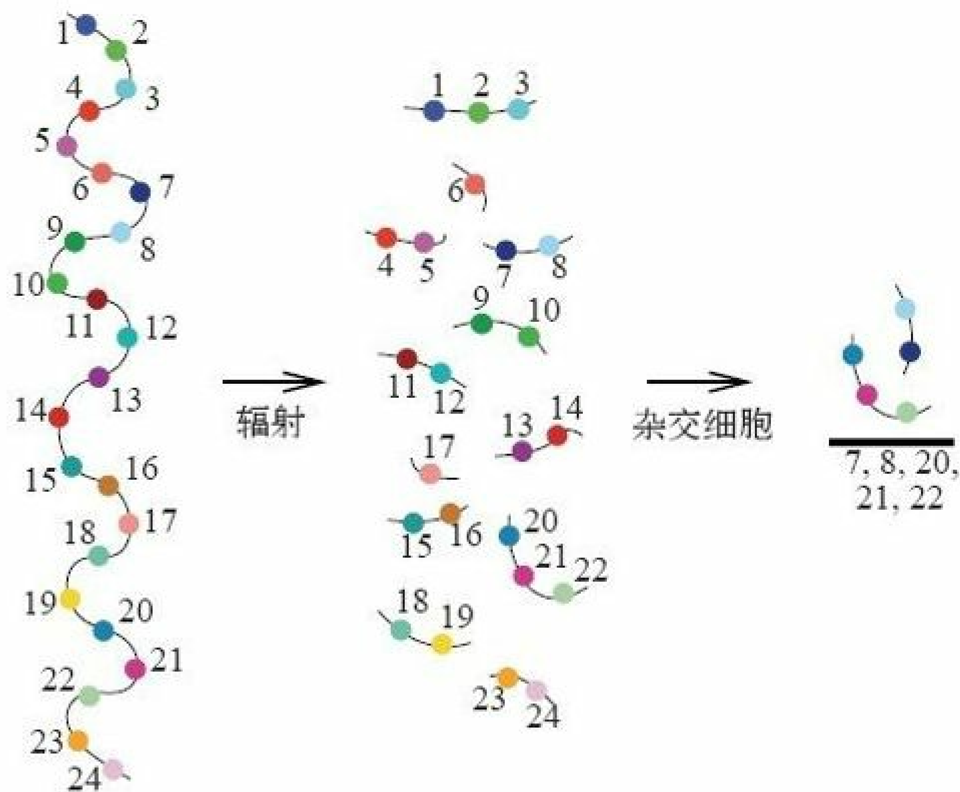


图**3-4**　辐射杂种细胞作图法原理

RH作图技术的核心思想是，通过分析标记两两之间能否并存于细胞系中，收集总结标记的位置信息。具体说来，如果标记A和标记B在染色体上距离很近，那么辐射就不太可能使染色体在它们之间断开，所以假如在一个细胞系里有标记A，那么很可能也有标记B。反之，如果标记A和标记B在染色体上距离很远，那么就可以预料，A和B将分别出现在不同的细胞系中，两个标记在某个细胞系中并存的可能性极小。研究人员可以巧妙利用这种推理方式，精心得出两个标记间距的实验值。有了两两之间距离的实验值，则测定染色体上标记顺序的问题就转化成为TSP模型。事实上，可以把这一顺序看作周游所有标记的哈密顿通路。这条通路很容易转化为回路，正如我们之前的做法那样，只需要增加一个额外的“虚城市”点即可。

在美国国立卫生研究院（National Institutes of Health），Richa Agarwala和Alejandro Schäffer带领一个小组研究这类基因组问题。针对这些具有实际背景的TSP，他们已经提出解决方法，开发了软件，还考虑到了如何处理错误数据（这属于实验室里的家常便饭）。[1]他们的软件使用了Concorde代码，保证能找出最优路线。这个软件为许多重要的

研究作出了贡献，比如用来构建很多物种的基因组图谱，包括人、恒河猴、马、狗、猫、小鼠、大鼠、奶牛、绵羊和水牛。

1. Agarwala, R., et al. 2000. Genome Research 10, 350–364.

3.3 望远镜、X射线、激光方向瞄准

虽然我们讨论TSP的应用时，一般考虑的情形都需要跨越遥远的空间距离，实际到达远方的地点，但是有些TSP题目产生的背景却不同，无需亲身上路，只是从远处观测目的地而已。通过某种望远镜观测行星、恒星和星系，就是一个很自然的例子。

为了观测，需要把望远镜设备旋转到合适的位置，这一操作称为快动（slew）。大型望远镜的快动由计算机驱动电机调节操控，整个过程复杂耗时。对于一组观测，有一条TSP路线可以让每次快动耗费时间之和达到最小，因此可以用来安排总计划。这道TSP里的城市也就是需要拍摄照片的观测对象，各地之间的旅行费用也就是望远镜在拍摄各个物体之间的快动过程所需的时间。

Shawn Carlson在《科学美国人》上发表了一篇文章，介绍了TSP的启发式解法如何帮他做计划，让一台望远镜每夜能拍摄大约200个星系，而这台机器年岁已久，经不起折腾。Carlson描述自己对TSP的好解法的需求时，这样写道："这台望远镜在整个天空内大幅摆动时，它的40岁高龄的驱动系统就会虚脱崩溃，因此绝对需要尽可能少移动这位年老体衰的老伙计。"[1]现代天文望远镜装置当然并不算年老体衰，不过要想高效使用这些极其昂贵的仪器，TSP的好解法确实是绝对需要的。

1. Carlson, S. 1997. Sci. Am. 276, 121–24.

3.3.1 搜寻行星

美国宇航局在筹划空间望远镜的工作任务时，考虑到了一些有趣的TSP题目。美国喷气推进实验室的Martin Lu称之为另一种TSP——traveling planet-finder problem（旅行行星搜寻者问题），因为空间望远镜的一大目的就是作为"行星搜寻者"，在太阳系附近寻找环绕其他恒星旋转的类地行星。

与地面望远镜类似，空间望远镜的TSP也是为了确定观测的先后顺序。不过这次，观测顺序不能像地面上那样每晚临时确定，而必须在任务开始之前很久就作出决定，因为空间望远镜的快动需要耗费大量燃料，而且研究每颗恒星也需要花费大量时间。据Martin Lu预计，一项为时3年的任务大约可以对50颗恒星完成观测。

对一颗可能是类地行星的星体进行观测时，会遇到困难，因为它附

近的恒星发出的强光会向镜头方向直射而来，导致拍摄这颗行星的任何照片都因过度曝光而呈现一片白色。一种推荐解决方案是配备一块大面积遮光板，安置在距离空间望远镜50 000至100 000千米处。普林斯顿大学的Robert Vanderbei介绍说，这就像在望远镜的眼前举起一只硕大的拇指挡住恒星的光一样。空间望远镜沿着固定轨道运行，遮光板则在不同的位置之间移动，从而实现观测。

在普林斯顿大学，Egemen Kolemen和Jeremy Kasdin对于依靠遮光板的空间望远镜进行了详细的研究。[1]他们利用一系列优化模型，估计遮光板在各颗恒星之间移动时耗费的燃料量。当使用两块遮光板与一台望远镜共同工作时，他们解得的结果如图3-5所示。望远镜将按照顺序轮流观测各颗恒星，红线和绿线分别代表两块遮光板的移动路线。注意图上有些路线片段看起来像是分离的，但实际是相连的，只不过连线在球面背面，图上看不出来。美国宇航局的目标名单上总共有100颗备选恒星，这次测试解得的路线经过了其中的80颗。

1. Kolemen, E., N. J. Kasdin. 2007. Adv. Astronaut. Sci. 128, 215–133.

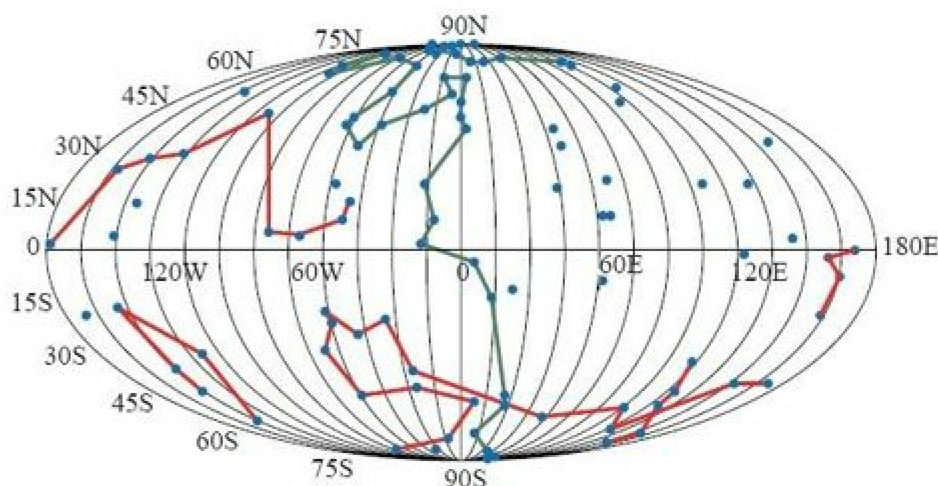


图**3-5**　观测**80**颗恒星时两块遮光板的移动路线（**E. Kolemen**和**N. J. Kasdin**供图）

3.3.2 X射线晶体学

在另一个截然不同的领域里，也出现了类似地面望远镜问题的TSP研究。20世纪80年代中期，Robert Bland和David Shallcross与康奈尔大学的一个研究小组合作，在X射线晶体学实验中，利用TSP的模型来操控衍射仪的移动。[1]计算机驱动电机需要移动晶体试样，还要重新让X射线

瞄准试样，有时对每个晶体样品需要进行多达30 000次测量。电机完成这些定位步骤所需的时间就是这道TSP题目的旅行费用。Bland和Shallcross报告说，借助TSP方法，他们最多可以缩短总定位时间的46%。

1 Bland, R. G., D. F. Shallcross. 1989. Op. Res. Let. 8, 125–228.

3.3.3 激光雕刻水晶工艺品

另一种方向瞄准类的TSP起源于制造业使用的脉冲激光。脉冲激光可以用来制作模型，或者在坚硬透明的水晶内部蚀刻图案，比如图3-6的pla85900摆件[1]，作者为Precision Laser Art公司的Mark Dickens。从激光的这类用途中产生了一道不错的TSP题目。激光束射向水晶内部，焦点使水晶的特定部位发生细微的爆裂，产生许多极小的白点，因此可以透过清澈透明的水晶看到这些点。TSP的作用就是设计激光经过各点的顺序，使制作所需时间最短。

1. 这里的pla85900参见1.3节图1-7，就是之前提到的那道包含85 900座城市的芯片TSP题目。——译者注



图**3-6**　激光在水晶内部雕刻的图案

把精美的美术作品临摹到水晶上时，为了得到高品质的图案，需要刻画的点非常多。为此，Dickens采纳了Concorde代码给出的启发式方法。如果TSP有光荣榜，这一应用必然榜上有名，因为它为TSP带来了若干工业实例，有些题目是有史以来规模最大的，包含的城市数甚至超过100万。

3.4 操控工业机械

在现代制造业中，像钻孔、装配这样的重复性工作常常由机械设备完成。以此为背景产生了很多TSP应用题。

3.4.1 印制电路板钻孔

常见电子设备中都含有印制电路板（printed circuit board，又称印刷电路板，简称PCB）。印制电路板上面经常有大量小孔，用来装载计算机芯片或者导通各层之间的电气连接。制作过程中，自动钻孔机在特定位置之间移动，从而依次打出这些小孔，找出钻头总移动时间最短的路线就是TSP的一大经典应用。Gerhard Reinelt的TSPLIB是旅行商问题的实验数据库，收集的案例中包括一些电路板钻孔类的题目，其中一道题对应的印制电路板如图3-7所示。

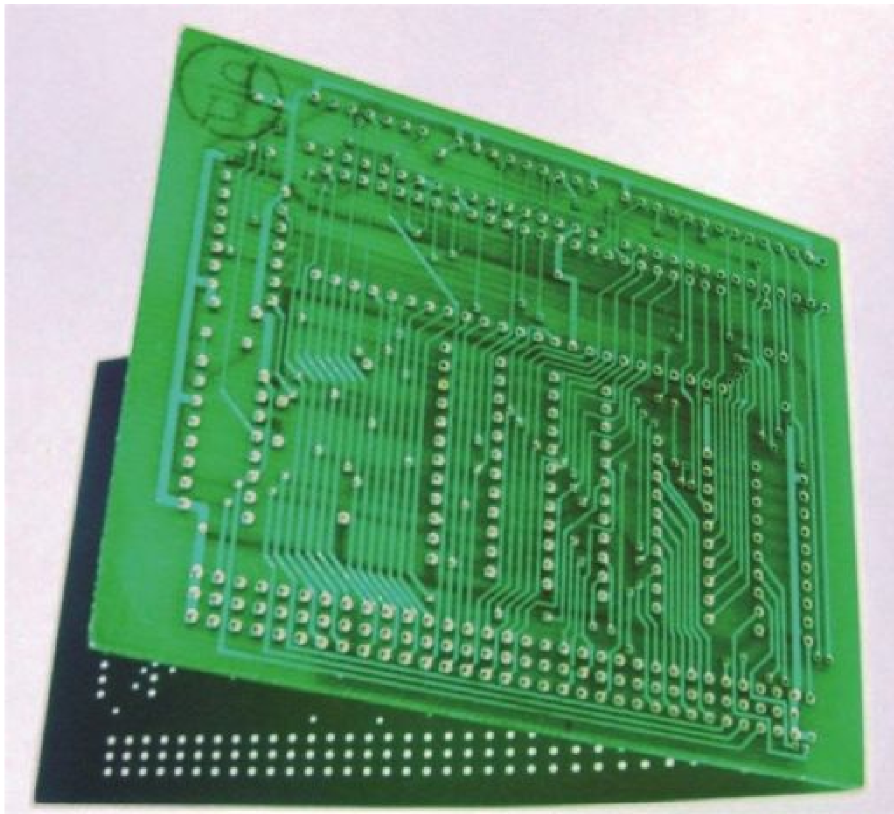


图**3-7**　有**411**个孔的印制电路板（**Martin Grötschel**供图）

应用TSP算法可以使电路板生产线的总生产量提高10%左右。[1]这一

类TSP题目的规模一般为几百至几千座城市。

1. Grötschel, M., M. Jünger, G. Reinelt. 1991. Zeit. Op. Res. 35, 61–84.

3.4.2 印制电路板焊锡

德国电子工程师Wladimir Nickel在文章中介绍说，他在电路板焊锡步骤中借助了Concorde程序。焊锡（solder）是印制电路板生产过程的一步后续处理，目的是把贴片元件焊接到电路板表面。Nickel使用一台装有锡膏点胶机的数控（计算机数字控制，简称CNC）机械，将锡膏涂布于电路板表面的特定位置。图3-8展示了他使用的数控机械。照片上他制作的电路板有256处位点需要点涂锡膏。Concorde程序帮他解出了这道TSP题目的最优路线，也就提供了点胶机走完所有位点的最快方式。

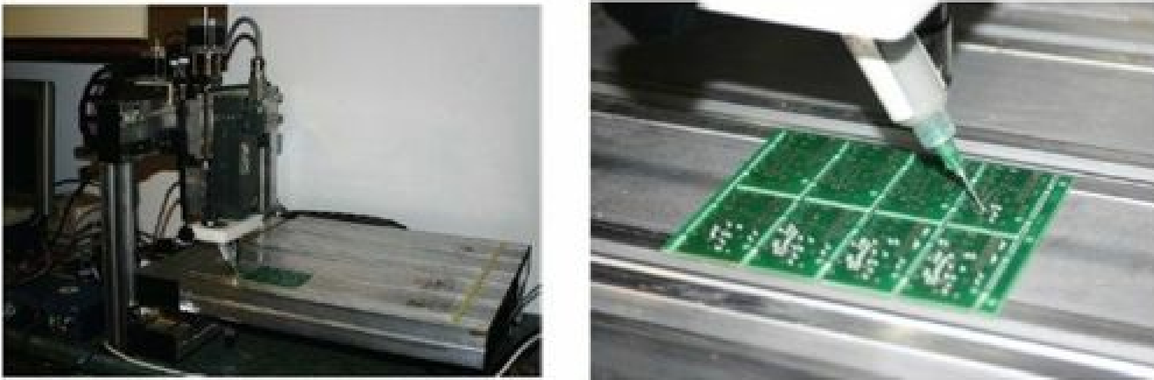


图**3-8**　印制电路板涂布锡膏（**Wladimir Nickel**供图）

3.4.3 黄铜雕刻

人们使用黄铜冲压模具制作浮雕图案，比如巧克力包装盒上凸起的装饰图。黄铜模具以前都是手工制作的，但现在一般由重型数控铣床铣削雕刻而成。数控铣床刻完一个字母或者其他设计元素之后，主轴升高，铣头移动，准备刻下一个字母或设计元素。由于每个设计元素都不只是单个小点，因此铣头移动到整个元素上方的任意一点都可以，这就引入了灵活变通的可能。2008年，数控雕刻师Bartosz Wucke写道，如果模具图案有大量字母或者是大量点纹组成的抽象图案，则应用TSP模型可以使加工时间减半。

3.4.4 定制计算机芯片

20世纪80年代中期，在贝尔实验室的研究工作中，涌现出了类似的TSP应用，不过这次的空间尺度要小得多。贝尔实验室的研究人员发展了一种快速生产定制计算机芯片的技术。生产原料是一块原始的芯片，一些称为逻辑门（logic gate）的简单组件在芯片上形成了网络。然后研究人员用激光切割网络中的部分位点，把这些逻辑门组件划分成各组逻辑门电路，分别实现指定的芯片功能。这里需要引领激光经过各切割位点，也就转化成为TSP题目。Johnson提出了更快的TSP启发式算法，使一般情况下的激光定向时间降低到原来的一半以下，实现了生产过程的大提速。

这项应用同样在TSP光荣榜上占有一席之地，因为创纪录的85 900座城市的TSP（见图1-7）就来源于此。

3.4.5 清理硅晶片缺陷

在激光切割步骤之前，计算机芯片生产过程中还会出现另一种TSP应用。硅晶片形状为圆形，面积比较大，由硅组成，而且必须不含任何杂质。设计的标准芯片要蚀刻到硅晶片上。纳米制造技术企业美国应用材料公司（Applied Materials）有一项清理硅晶片缺陷的技术。为了操控机器在各个缺陷之间移动，他们也用到了Concorde程序。

3.5 组织数据

数据挖掘是从大量数据中寻找规律的技术。一种基本的数据挖掘工具称为聚类（cluster），就是组织信息，把性质相似的信息分到同一组里，这种组称为簇。有时，每两个数据点之间的相似度都可以具体表示，这时就会用TSP来处理信息。把两点之间的相似度数值作为旅行费用，数据点关系越密切，相似度数值就越高，因此总费用最大的哈密顿通路就会使相似点彼此接近，通路的片段也就可以作为备选的簇。然后通常由研究者人工完成最后的分类。他们选出序列的自然分界点，从而在这些点处把通路分成几段。[1]

1. Lenstra, J. K. 1974. Operations Research 22, 413–414.

上面这种两步走的方法可以被另一种简洁的技巧取代。新方法的提出者是Sharlee Climer和Weixiong Zhang。[2]他们在解决这道TSP题目时，添加的“虚城市”个数不是1，而是k+1。每一个“虚城市”与其他任何城市之间的旅行费用都是0。由于不同簇的两点之间费用较低，而相同簇的两点之间费用较高，因此一条好的TSP路线就会把“虚城市”插入在不同簇的数据点之间，尽量避免降低总费用。所以，k+1个“虚城市”恰好可以作为k个簇的分界点。

2. Climer, S., W. Zhang. 2006. J. Mach. Learn. Res. 7, 919–943.

Climer和Zhang使用这种“TSP+k”方法进行基因表达数据聚类分析，利用Concorde代码计算最优路线，通过改变k的值研究不同的簇数目对数据分析的影响。他们开发的软件可以得到图3-9这样的图像。图中的数据集是拟南芥（Arabidopis）的499个基因在五种不同环境条件下的表达情况，灰色阴影代表基因表达水平数值，白色实线则表示各个簇的分界线。

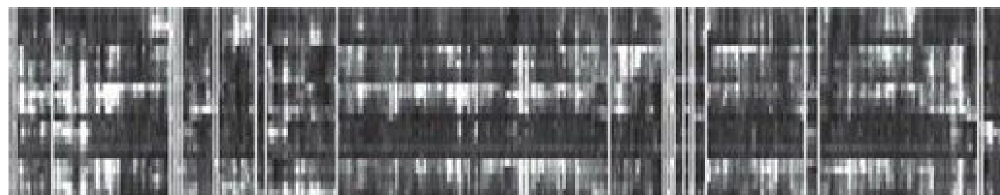


图**3-9**　基因表达数据（**Sharlee Climer**和**Weixiong Zhang**供图）

3.5.1 音乐之旅

在计算机音乐领域，人们同样借助TSP来理解大量由计算机编码的

音乐。Elias Pampalk和Masataka Goto（后藤真孝）在日本产业技术综合研究所工作。用户借助他们发明的MusicRainbow（音乐彩虹）系统，可以找到适合自己音乐品味的其他艺术家（音乐人）。Pampalk和Goto收集了558位艺术家演绎的15 336首曲目。他们首先通过计算比较这些曲目的音频属性，建立了一种测定艺术家相似度的方法。然后，他们使用TSP模型，用一条环形路线串起所有艺术家，使彼此相似的艺术家在路线上的次序也比较接近。模型中的城市就是艺术家，而两地之间的旅行费用就是两人之间的相似度。

有了这条环形路线，用户就可以通过旋转一个旋钮在音乐库中进行选择。MusicRainbow设备配有计算机屏幕，用来显示艺术家的信息。屏幕上有一系列彩色的同心圆，分别对应了音乐的不同高级分类，比如摇滚（rock）和爵士（jazz），从而表明了艺术家的种种标识属性。MusicRainbow系统的一大亮点在于，所有标识信息都是自动搜索网页获得的，因此任何音乐库都可以很方便地使用该系统。

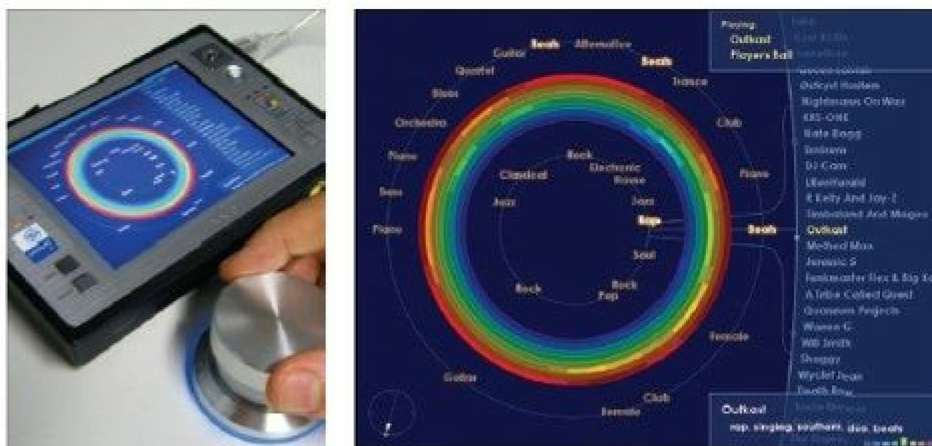


图**3-10　MusicRainbow**设备（**Elias Pampalk**供图）

Elias Pampalk还曾经参与另一项音乐方面的TSP应用项目。另外两名合作者是奥地利林兹大学的Tim Pohle和Gerhard Widmer。他们的目的是把一批音乐曲目组织起来，整理出一个循环链表，让相似的曲目在链表上位于邻近的位置。这样一来，用户使用音乐播放器时，可以通过旋转一个转盘，选出一首契合自己当下心情的曲目。播放器放完这首曲目之后，便会接着播放一系列类似的曲目。这三位研究者设计的播放器名为Traveller's Sound Player（旅行者的音频播放器），通过比较两首曲目的音色相似度，来测量两者之间的“距离”。他们使用了3000多首曲目作为一个测试用例，并用TSP模型求出了总距离最短的环形路线排列次序，得到了相应的循环链表。

上面这两个例子的涉及面都比较广，但是也有范围比较局限的应用实例。纽约大学的Drew Krause在自己作曲时借助了TSP的力量。在音乐中，流畅平稳的过渡称为级进旋律（conjunct melody），能给人带来愉悦之感。Krause在安排和弦时使用Concorde代码，就是为了让每个和弦转到下一个和弦的变化幅度都尽可能小。在这个TSP模型中，城市是他要用到的一组和弦，两点间的旅行费用则定义为两个和弦的对应音符之间相差的半音数之和。

3.5.2 电子游戏速度优化

为了表现出木材、金属等物质材料的真实质感，现在的新型电子游戏在显示物体时都需要使用大量数据。这种显示数据的基本组成部分称为纹理（texture）。目前，有的纹理库包含成千上万种纹理，从砖石到铁锈，各种纹理应有尽有。在游戏里，为了描绘显示的所有物体的表面，任意一幕场景都需要一组特定的纹理，也就是一个纹理组。由此产生了一大挑战，即如何尽快获取纹理数据并在机器屏幕上显示出来，从而使场景切换时可以流畅过渡。这里正是TSP的用武之地。

DVD代表数字视频光盘（digital video disk）。[1]DVD的数据读取速度与数据存储方式有关，这是它的一个基本特性。具体说来，读取顺序存储的数据远远快于乱序存储的数据。因此，纹理数据在光盘上的存储位置布局会严重影响显示游戏场景所需的时间。要是能让游戏各场景对应的各个纹理组按照顺序存储就好了，但是要想实现这种美梦，重复用到的纹理就必须储存好几个副本。另一种思路则是恰当选择纹理数据布局，使分隔（break）的总数尽可能小。这里的分隔指的是一组纹理分散在不同位置的情况，具体说来，如果这组纹理分布在光盘的*k*个区间内，那么就会产生*k*-1个分隔。回忆一下之前提到的绘制基因组图谱的例子，应用的方法是一样的，只需要把细胞系换成纹理组就可以了。这里的TSP模型中，城市代表不同的纹理，两点之间的旅行费用代表只包含两个纹理之一的纹理组数。正如基因组图谱的情形那样，这次同样不能直接得到环形路线，只能得到哈密顿通路，再用我们的老办法，增加一个“虚城市”把通路转化成回路。

1. 1995年正式确立DVD规格时，DVD被重新定义为digital versatile disk，即数字多功能光盘或数字通用光盘。——译者注

加拿大企业Digital Extremes的Glen Miner阐述了这项应用。这家公司使用Concorde程序求解纹理数据储存布局，做了相关的试验，并公布了

TSP模型带来的显著改进。

3.6 微处理器测试

英伟达（NVIDIA）是一家计算机技术公司。近年，该公司在测试图形处理器时，也采用了Concorde代码，用来优化测试时使用的集成电路。在设计新型计算机芯片时，制造后的测试步骤在整个生产过程中至关重要，因此TSP的此类应用非常普遍。

为了方便集成电路测试，人们于20世纪80年代引入了扫描链（scan chain）的概念。计算机芯片的元件称为扫描点（scan point）。扫描链是一条连接扫描点的通路，其两端位于芯片边界，分别为输入端和输出端，如图3-11所示。从芯片扫描链的输入端开始，各个扫描点依次加载测试数据，经过一系列测试后，可以在输出端读取并评估数据。

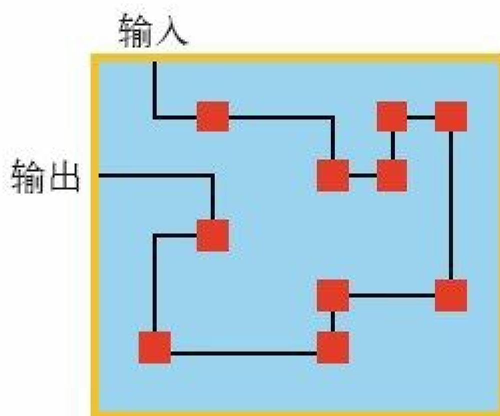


图**3-11**　扫描链示意图

人们使用TSP来确定测试点的次序，使测试链尽可能短，从而有助于实现许多目的，比如节省宝贵的芯片布线空间，再比如通过加快信号传输速度来节约测试阶段的时间。

绝大多数情况下，芯片制造技术有限，电气连接只能选择水平布置或者垂直布置。这意味着在扫描链的TSP题目中，两点之间的路程不是直线距离，而是连接它们的水平或垂直路径的长度之和，就像走在曼哈顿横平竖直的街道上那样。图3-12给出了一道扫描链TSP的最佳路线。这道题目包含764座城市，原题由太阳微系统公司的Michael Jacobs和Andre Rohe提供，答案则由Concorde程序解出。为了缩短测试所需时间，新型计算机芯片一般都会有多条扫描链。以这道764座城市的TSP为例，它代表一条扫描链，而这条扫描链所在的芯片共有25条扫描链，

它只是其中一条。

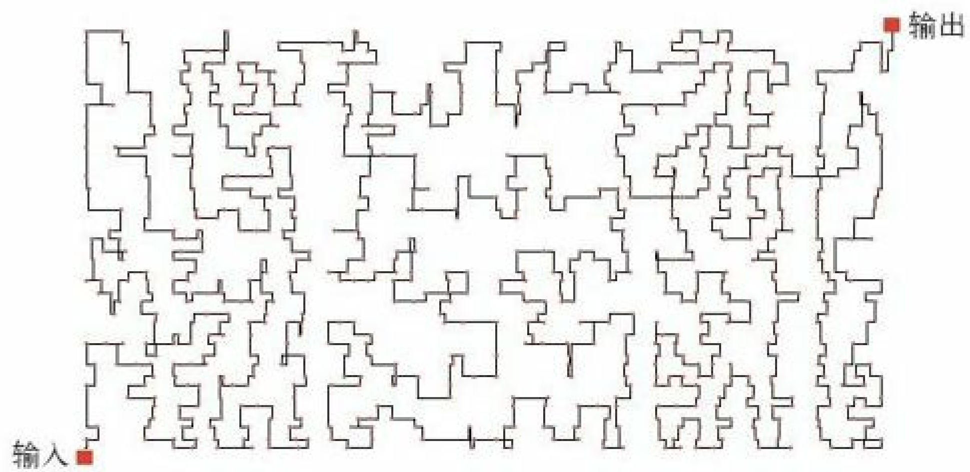


图**3-12**　有**764**座城市的扫描链**TSP**

3.7 安排生产作业任务

德国企业BÖWE CARDTEC（博威卡技术公司）为管理智能卡（信用卡、身份证等）生产过程提供软硬件产品。该公司的客户通常都用同一硬件设备生产多种不同类型的智能卡，因此在两轮任务不同的生产过程之间需要有重新配置的步骤，例如更换色带颜色并插入正确的空白卡。两轮生产作业之间的准备时间相当长，从而导致整体日产量下降。为解决这一问题，BÖWE CARDTEC软件使用TSP模型，把所有作业任务按一定顺序排列起来，使准备时间总和最小。城市就是这些作业任务，作业i和j之间的旅行成本则是硬件设备在完成作业i和开始作业j之间重新配置所需的时间。该公司报道称，应用Concorde解得的路线，通常情况下总准备时间的降幅最多能达到65%，从而总生产率也大大提高。

1954年，Merrill Flood在一次讲座中最早提出了TSP在安排计划方面的此类应用。在一般的例子中，从作业i到作业j的准备时间不同于从作业j到作业i的准备时间。因此这种TSP题目是不对称的，路线的总花费与旅行方向有很大关系。

3.8 其他应用

旅行商涉足了众多领域，旅行商问题的应用十分广泛，本章的介绍自然只是管中窥豹而已。其实在应用数学文献中，隔三差五就会涌现出TSP模型的新奇应用。下面列出一部分已取得成功并公开发表的应用课题。[1]

1. 参阅书目：Applegate et al. (2006), Section 2.7.

- 规划自然公园的徒步路线
- 尽量避免浪费壁纸材料
- 规划在长方形仓库里一次拿取若干货物的路线
- 在玻璃工业中优化切割玻璃的方案
- 设计能连接一切DNA片段的万能连接DNA
- 估算连接一组望远镜阵列所需的电缆挖沟费用
- 研究与演变过程相关的问题
- 根据已有的片段序列数据库整合绘制基因组图谱
- 在地球物理学中收集地震数据
- 对只包含0~1数组的数据集进行压缩

从真正的旅行商计划路线问题，到这些旅行商问题五花八门的应用背景，两者相差何止十万八千里！

第4章 探寻路线

> 我们并不宣称该程序万无一失，只是说它能在可行的计算时间内给出较好的结果。

——Robert Karg和Gerald Thompson，1964年[1]

1. Karg, R. L., G. L. Thompson. 1964. Management Science 10, 225–248.

就算听到TSP不可解的断言，巡回旅行商也不会大惊小怪。他照旧会发动汽车，上路拜访顾客，履行自己的使命。基于这种脚踏实地的心态，便出现了另一种思路，即暂时抛开必须找到绝对最优解的念头，转而关注如何尽快求出近优解。在新观点的影响下，人们为了让旅行商来得及回家吃晚饭，提出了各种富有想象力的想法。事实上，在这一方向的TSP研究中，许多技巧和方法得到了发展和应用，如今亦成了计算科学领域的重要工具，如模拟退火算法（simulated annealing）、遗传算法（genetic algorithm）、局部搜索算法（local search）等。寻找路线的题目相当于沙箱，用来测试旨在从大量情形中寻找最优解的算法。这些题目同样也是TSP研究的训练场，只不过在这里，训练科目不胜枚举，得到的结论也非同小可。

4.1 周游48州问题

这道难题流行于20世纪40年代，题目是给一名旅行商设计路线，让他以华盛顿特区为起点和终点，周游美国本土的48个州。Julia Robinson建议把各州首府指定为旅行商的目的地，从而限制了路线的范围。尽管只需把各段路程的数值列成表格，就可以彻底明确题目叙述，但是当初似乎没有人做这一步工作。很可能是因为，在当时的人们看来，如此大规模的TSP题目超出了他们的解决能力。

显然，Dantzig、Fulkerson和Johnson并不这么认为。由于无法获得城市间路程的标准数据集，他们自己动手给出了一组数据。由图4-1可见，他们选择的城市与前人的限定大不相同，在分别选自各个州的48座城市中，只有20座是所在州的首府。这种背离传统的选择却并非出于什么神秘的原因。“我们之所以选择这组城市，是因为很容易在地图集里查出其中大部分道路的里程。”[1]有道理。这一选择尽管理由简单，但在TSP计算中，确实直接带来了好的开始。因为在他们使用的兰德麦克纳利标准地图集中，从华盛顿到波士顿的最短路线会经过美国东北部的另外7座城市，这些城市也在旅行商的目的地之列。于是，他们决定不考虑这7座城市。这样做有点冒险，论证过程如下：在周游剩下42座城市的最优路线里，如果有一段是从华盛顿直接开车前往波士顿，那么他们只需要在路过巴尔的摩、威尔明顿、费城、纽瓦克、纽约、哈特福德和普罗维登斯的时候，摇下车窗挥手示意就可以解决原题了；但是另一方面，周游其他城市的最优路线也可能以另外的方式到达华盛顿，若真如此，就只能换种思路从头再来了。

1. Dantzig et al. (1954).



图**4-1**　周游**48**州问题里的各大城市

看看地图，很容易猜到，最优路线大概会用到直接连接华盛顿和波士顿的这条路。实际确实如此。因此，兰德公司的Dantzig等人确实有理由只考虑剩下的目的地。然而有必要指出，放在今天就没有这么顺利了。通过谷歌地图（Google Maps）可以测得，从华盛顿到波士顿的最短路程是451英里（756千米），但是假如要求经过那7座城市，这段路就变成491英里（790千米）了。同样是从康涅狄格州一路开入马萨诸塞州，前者之所以能节省路程，主要是因为用到了84号州际高速公路。但是这段高速公路直到1967年才通车。

Dantzig等人当时由兰德麦克纳利地图查得的数据是对称的，城市两两之间的距离与行车方向无关。（在本章中，我们假定旅行成本是对称的。[2]）他们处理了初始数据，把每段距离值先减去10，再除以17，然后四舍五入，取最接近的整数。“我们选择这样的变形，是为了把原始表格里的d_{ij}数据都变成小于256的整数，以压缩用二进制形式存储距离表所需的空间，实际并没有什么用。”[3]他们的论文列出了处理后的距离数据表，从而明确提供了这道题目的条件。

2. 本书重点始终是对称形式的TSP，也就是旅行成本（路线长度）与旅行方向无关的情形，即从城市A去城市B的旅行成本与从城市B返回城市A相同。这一限制主要是为了控制讨论的篇幅，但是请不要认为作者有意蒙混过关，因为可以证明：任意包含*n*座城市的TSP题目都可以转化为2n座城市的对称TSP题目。

3. Dantzig et al. (1954).

钉子和绳子

Dantzig等人的数据并没有使用两点之间的真实距离，因此稍微改变了问题原有的几何学本质。不过，如果数据取为两点之间的直线距离，也就是欧几里得距离（欧氏距离），可以有效提高比较路线长度的效率。事实上，他们当时的解法就完全建立在这种近似的基础之上。

他们在原始论文里给出了一条周游美国的路线，对得到这条路线的方法却只字未提。随后，Dantzig在讲座中透露，研究小组当时使用了一个实物模型。模型是木质的，由他们自制而成，上面标有题中的49个地点，每处都有一枚钉子。代表出发地的钉子上系了一根细绳。他们把这根绳子依次绕过其他钉子，便勾画出一条路线。



图**4-2**　根据钉绳法构造周游德国的路线（德国柏林祖斯研究院供图）

Dantzig表示，这个模型是他们手工解题的得力助手。绷紧绳子测量绳长就可以获知路线的长度，也能很快直观看出下一段子路线的可能走向。这个模型完全没有提供解题的算法，却帮助Dantzig等人确定了一条周游所有城市的路线。并且他们后来证明，这条路线就是所求的最优路线。按照处理后的数据表来计算，周游美国的最优路线长度为699个单位长度，换算成地图集上的实际距离则是12 345英里（即19 867千米）。

4.2 扩充构造树与路线

无论是在白纸还是在木制模型上勾画路线，思路都有一点别致之处：先选取一个出发点，然后逐步扩展出一条路线，扩展的过程可以理解为依次加入其他地点，也可以理解为依次加入其他路段。Dantzig等人凭借直觉决定每次应该选择哪枚钉子，但简单的算法也可以像样地完成任务。

4.2.1 最近邻算法

在从头开始构建路线的时候，最容易想到的方法就是每次都在没有到过的城市中选择最近的一个。这种算法称为最近邻算法（nearest-neighbor algorithm），看起来合情合理，然而得到的路线通常不是最短的。

对于42座城市的删减版题目，根据Dantzig等人提供的距离数据，可以按照最近邻算法求出一条路线。分步过程如图4-3所示。这条路线从菲尼克斯（Phoenix，凤凰城）出发，很快在美国南部延伸开来，前面若干步看起来都很不错，但是到了太平洋西北地区以后就会发现，除了一路回到东海岸以外别无选择，因此接下来只能把之前满不在乎地漏掉的城市一个一个补回来。使用最近邻算法的后果一贯如此，前进的时候目光短浅，最后总会把自己逼入死角。图4-3最后得到的完整路线长度为1013个单位长度，而Dantzig等人得到的最优路线仅为699个单位长度。

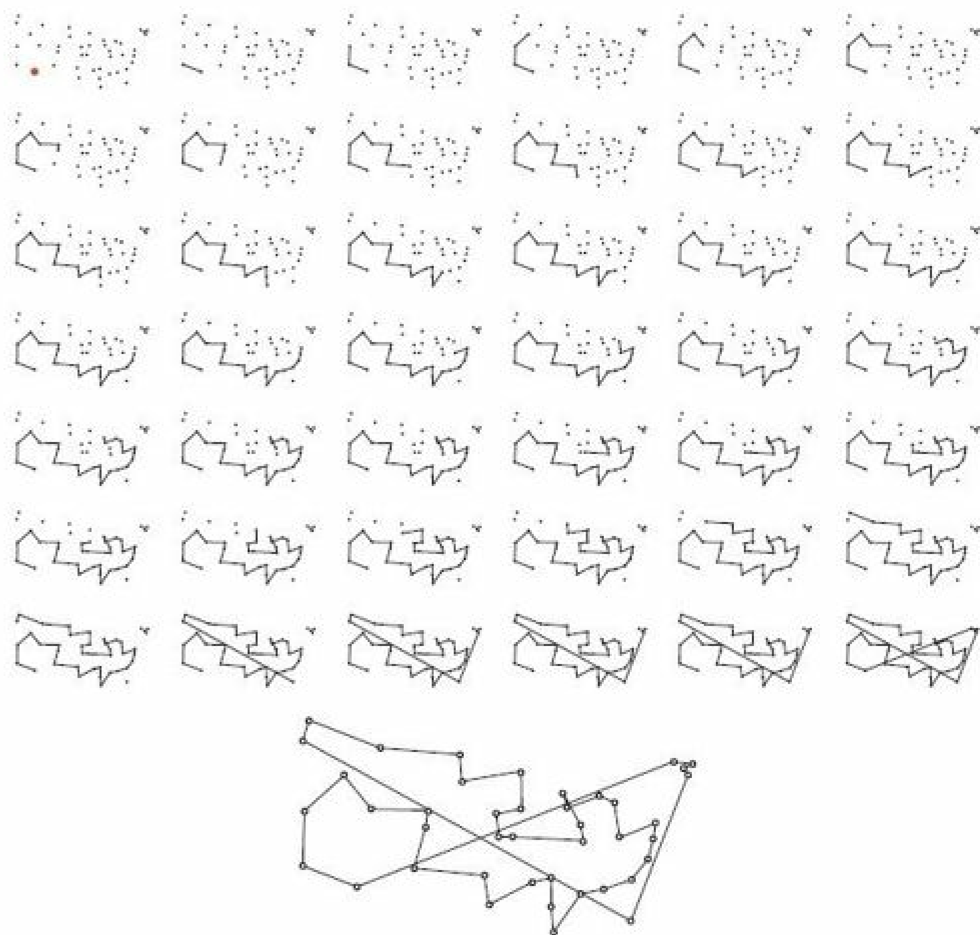


图**4-3**　基于最近邻算法构造路线

如果你想故意抬扛，也很容易设计一道TSP的题目，让最近邻算法得到的路线非常长，和最优路线相比简直糟糕透顶。重点就是，旅程的最后一程必须从最后一座城市回到出发地，无论这条路有多远都别无选择。所以，如果让蒙彼利埃和菲尼克斯之间的距离增加1 000 000个单位长度，很不幸，最近邻算法还是会得到同一条路线，而且这次的长度为1 001 013个单位长度，而最优路线不受影响，依然是699个单位长度。

上面这种修改方式比较无耻，得到的TSP题目虽然在数学上是合乎规定的，但其旅行距离的情形却违背了实际的道路情况。实际上，任何合理的TSP题目都应该满足三角不等式：对任意三座城市A、B、C，从A到B的路程加上从B到C的路程应该大于等于从A到C的路程。有了这条限制，上面那种没素质的题目就被排除在外了。事实上，对于*n*座城市的TSP题目，若其旅行成本对称且满足三角不等式，则可以证明最近邻路线成本不会高于最优路线成本的1+log(n)/2倍。因此，当城市数为50时，最近邻路线必然不超过最优路线的4倍；城市数为一百万时，最近邻路线不超过最优路线的11倍。你如果指望用最近邻算法做旅行计划，

可能看到这里还是不放心。没关系，接下来介绍的其他算法可以保证更好的结果。

1. 本书用到的对数（log）均以2为底。

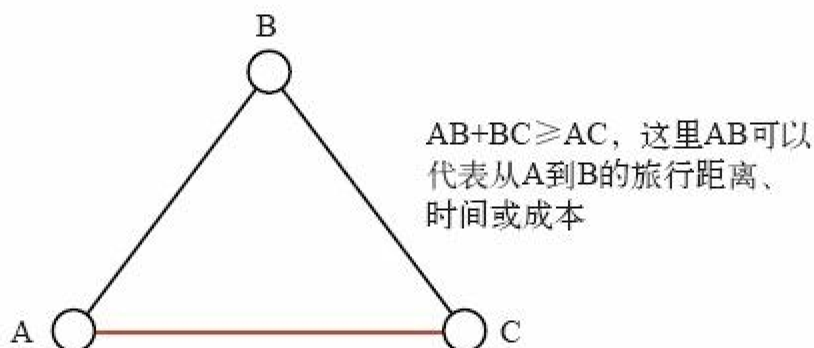


图**4-4**　三角不等式

4.2.2 贪心算法

最近邻算法只会生成一条路线。这条路线四处曲折延伸，最终遍历所有城市，其中每一步都选择最短的一段路程，因此这种算法非常贪心，但是“贪心”一词却被用来命名另一种算法[1]。贪心算法（greedy algorithm）同时生成许多段子路线，每一步都把最短的可用路线片段加入解集中。同样以环游美国问题为例，图4-5中给出了贪心算法求解路线的分步过程。每次新增的子路线位置分散在地图各处，但最终连成了一条完整的路线。

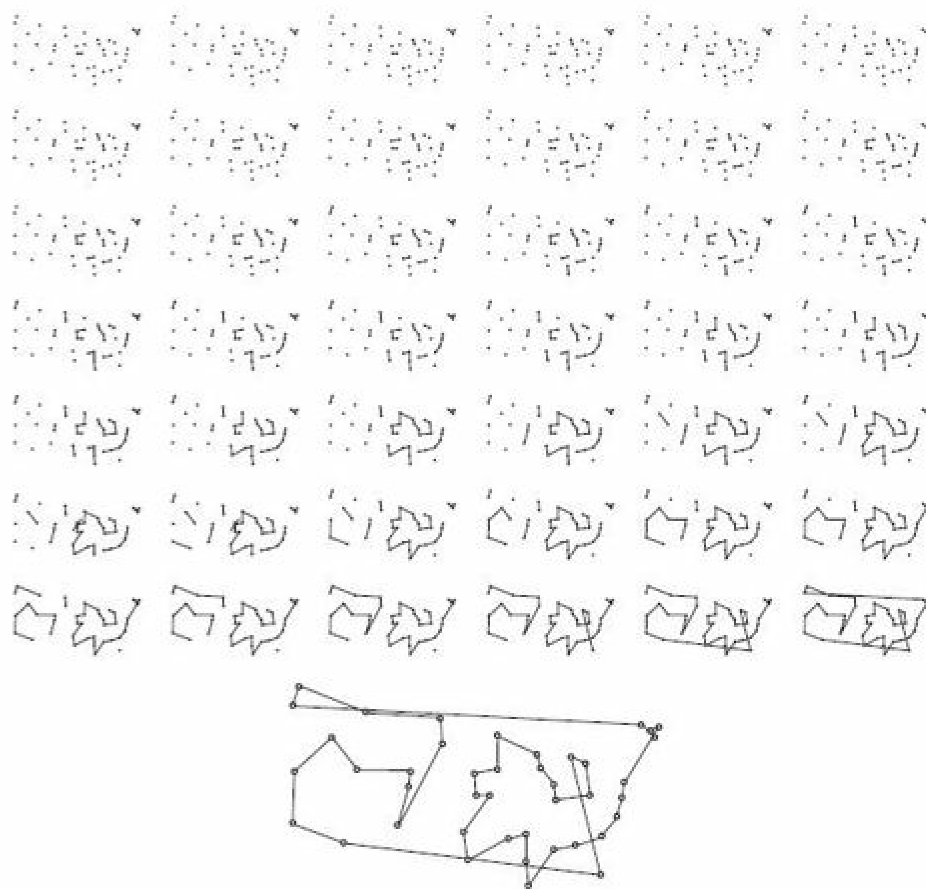


图**4-5**　基于贪心算法构造路线

1. 最近邻算法也是基于贪心算法的思想，因此有时它也称为贪心算法的最近邻点策略，而这里介绍的贪心算法则称为贪心算法的最短链接策略。——译者注

用图论语言可以很方便地描述贪心算法这样的TSP解法，其中图的顶点就是城市，图的边则是城市之间的路线片段。最终路线是图中一条哈密顿回路，选择的边与旅行商实际出行的路段相互对应。

贪心算法以最短边优先为原则。如果一条边不能连接两条子路线，形成更长的子路线，就不会把它加入解集。按照这个算法求解路线时，起初会让人莫名其妙。在环游美国的例子中，大约前20步生成的边都确实相当短。问题出现在求解过程的最后几步，因为这时我们为了把子路线全都连接起来，不得不接受几条非常长的边，结果导致总路程长达995个单位长度。

如果测试范例很大，贪心算法几乎总是明显优于最近邻算法。试举一例，若城市在正方形内随机分布，以直线距离作为每段路程，则贪心算法得到的路线长度一般不会超过最优解的1.15倍，而最近邻路线长度则通常不超过最优解的1.25倍。遗憾的是，这个数据只是实验观察得到的。目前已知在最差情况下，对于满足三角不等式的题目，贪心算法的

路线长度只能保证不高于最优解的1/2+log(n)/2倍，因此只是比最近邻算法好一点点而已。

4.2.3 插入算法

1954年，Dantzig等人借助钉绳模型获得了环游美国问题的最优路线，但是进一步的研究工作不可能指望用这种方法取得最优路线，所以当时人们面临一个紧迫的问题：他们的成功与钉绳模型的关系有多大？第二年夏天，年轻的兰德公司研究员John Robacker为此投身TSP研究，由随机路线出发，用Dantzig等人的方法，做了一系列试验，成功解决了好几道包含9座城市的题目。这些题目规模太小，说服力不强，但他还描述了通用的路线求法。如果题目的数据集很大，这种方法可以实现自动化。[1]

1. Robacker, J. T. 1955. RAND Research Memorandum RM-1521.

> 根据这些试验研究，A. W. Boldyreff提出了求近优路线的一系列步骤。他的方法具有两大优点：原理简单，使用快捷。以环游美国49座城市的题目为例，最优路线为699个单位长度，而这种近似算法求出的近优路线则为851个单位长度，误差为20%。

这种算法的思路是从一条周游几个城市的子路线出发，逐个增加新的城市，让路线像橡皮筋一样扩展，从而包围更多的城市。

由Boldyreff和Robacker提出的技巧，可以得出一类方法，统称为插入算法（insertion algorithm）。按照选取城市的具体规则，插入算法可分为几种，分别为最小插入法（cheapest）、最近插入法（nearest）、最远插入法（farthest）、随机插入法（random）。无论是哪一种插入算法，在插入新城市时，都会选择路线上最合适的位点，使插入后的路线长度最短。

最小插入法就是Robacker介绍并试验的方法，每次选择城市的原则都是使插入后的路线长度最短。最近插入法每次比较所有剩余城市到当前子路线上每一座城市的距离，选择相距最近的一对城市，其中，不在当前子路线上的城市就是要插入的。最远插入法则相反，每次选择的城市都是距离当前子路线上任意城市最远的一座。而随机插入法每次都是随机选择，把剩余城市中任意一座插入到子路线中。

在这些算法中，我最喜欢最远插入法，因为它一开始就能把握路线的整体形状，在插入城市的过程中也会不断完善细节。以环游美国问题

为例，图4-6给出了最远插入法的分步求解过程。出发点是菲尼克斯，前五步就扩张到新奥尔良、明尼阿波利斯，还有两个波特兰市[2]，然后逐渐构造出一条778个单位长度的路线。

2. 美国有两座城市都叫波特兰，分别位于美国西北部（属于俄勒冈州）和东北部（属于缅因州）。——译者注

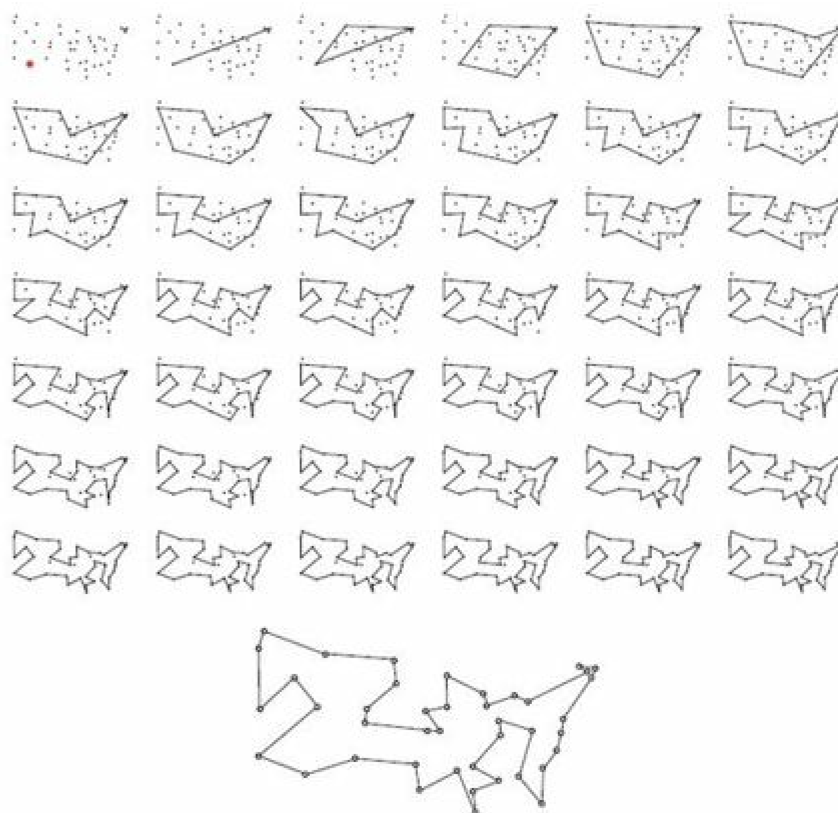


图**4-6**　基于最远插入法构造路线

研究表明，在满足三角不等式的条件下，最小插入法和最近插入法都相当不错，得到的路线长度不会超过最优路线长度的2倍。[3]虽然在实际应用中，最远插入法往往效果最好，但奇怪的是，在理论上，其路线长度只能保证不超过最优路线的log(n)倍。

3. Rosenkrantz, D., et al. 1977. SIAM J. Computing 6, 563–581.

4.2.4 数学概念：树

最近邻算法和贪心算法往往虎头蛇尾，最初的选择似乎很合适，最终的路线却总是令人失望。对旅行商来说，贪心没有好结果。然而出人意料的是，对于一道与TSP相关的题目，确实有一种贪心算法能保证得到最优解。这道相关题目是给定一组城市，要求选出一系列道路，能够

连接所有城市，并且总成本（代价）最小。对于环游美国问题的数据集，最小代价的道路结构如图4-7所示。道路总长度为591个单位长度，因此比最优路线还短不少。

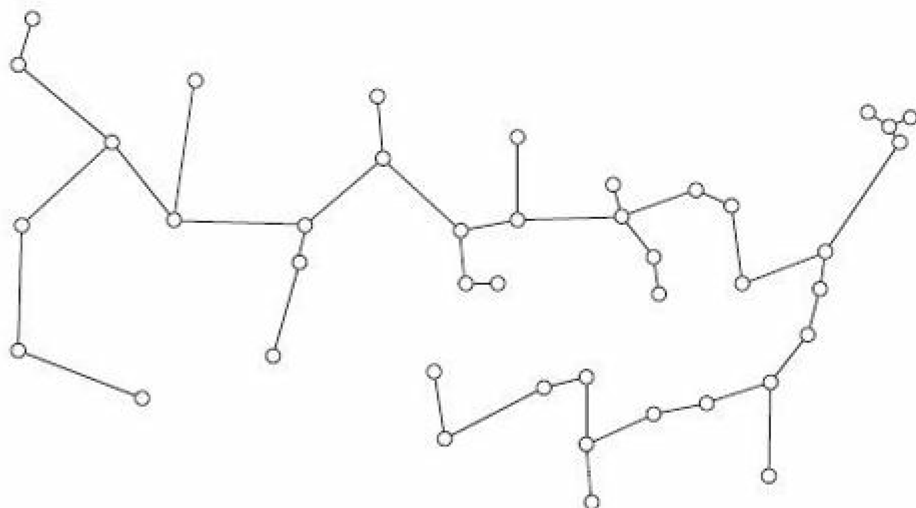


图**4-7**　最优树

我的曾曾曾曾曾祖师爷Arthur Cayley研究过这样的图。观察图4-7，请注意，这种结构是连通的，而且不包含回路。Cayley给此类图起了一个合适的名字：树（tree）。同时，他还把图的顶点称为“节点”（knot，该词也可表示“树的节疤”）。因此他的数学著作读来颇有植物学韵味：“在一棵有*N*个节点的树中，随意选择一个节点作为根节点，便可以认为这棵树是从根节点生长而成的，故称为有根树。”[1]我们这里并不介绍如何从根节点生成一棵树，而是利用树的结构给出TSP的一种解法，并且同样可以保证这种解法得到的路线长度不超过最优路线的2倍。顺便一提，在电影《心灵捕手》中，马特·达蒙（Matt Damon）饰演的男主角解决了一道神秘的数学题，那道题目就是关于树的。图4-8是影片中的一幕，他写了几行字，描述了用来计数包含*n*个顶点的树的Cayley公式，还画出了几棵顶点较少的树。

1. Cayley, A. 1881. Am. J. Math 4, 266–268. 原文为“In a tree of N knots, selecting any knot at pleasure as a root, the tree may be regarded as springing from this root, and it is then called a root-tree”，其中“knot”有“节点”和“树的节疤”两层意思，“root”也有“根节点”和“树根”两层意思，故有此言。——译者注



图**4-8**　马特·达蒙在电影《心灵捕手》中的形象，版权归米拉麦克斯影业公司（**Miramax Films**）所有

对于那个相关的道路连接问题，最优解确实就是一棵树。你大概已经凭感觉相信了这一点。关键在于，在建设道路网的时候，绝对不能形成回路，否则回路的最后一条边就是冗余的，因为它连接的两座城市之前已经通过其他道路连接起来了。这种情况下，使用贪心算法时，应该以最短边优先为原则。如果通过已经选择的其他边，可以从一条边的一个端点到达另一个端点，那么就不能把这条边加入解集。按照贪心算法，在求解过程中，各点会逐渐连接起来，形成分散的几大部分，各部分规模越来越大，最终生成一棵包含所有城市的树。不难证明一个出色的结论：根据这种简单的方法得到的树一定是最小代价生成树，也就是说总成本一定是最小的。[2]

2. Kruskal, J. 1957. Proc. Am. Math. Soc. 7, 48–50.

树并不是环形路线，但是可以给出周游各城市的方式，下面给出一种做法。在一棵树中，到达一座新城市后，如果有一条边从这座城市出发，尚未经过搜索，便沿这条边继续前进，到达另一座新城市；否则，如果从这座城市出发的所有边都已搜索过，则逐个回溯之前访问的城市，直到发现某座城市连接的某条边未经搜索为止。重复上述过程，最终会到达所有城市，并回到出发点。这样的路线称为按照深度优先搜索算法（depth-first search）遍历（traversal）一棵树。

深度优先搜索过程如4-9所示。图中的树有6个顶点（城市），虚线表示未经搜索的边，单实线表示经过一次搜索的边，双实线表示经过搜索和回溯的边。请注意，在结束时，每条边都刚好走过两次，这就意味着旅行路线的成本是树的代价的两倍，而最优树的代价不可能超过最优路线的成本，因此这个结果很不错。要想根据遍历路线得到一条实际路线，只需跳过回溯步骤，抄条近道即可。在图4-9的最终路线中，这些

近道以红线表示。

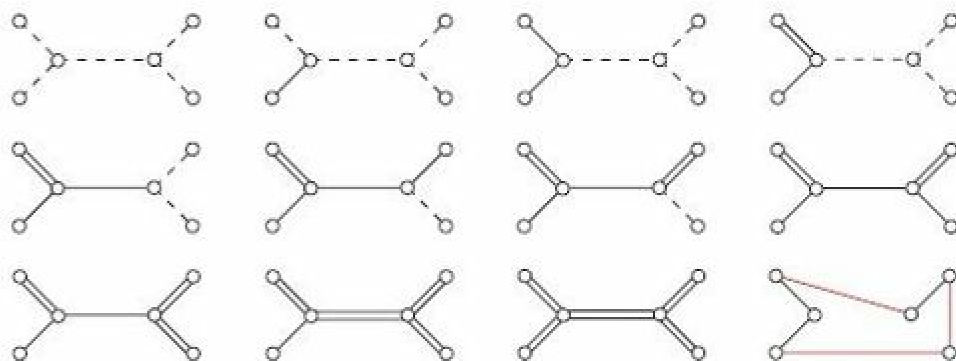


图**4-9**　根据树的遍历构造路线的步骤

按照这种算法，可以构造出环游美国问题的路线，长度为823个单位长度。如图4-10所示，深度优先遍历的起点是菲尼克斯。搜索过程中，如果从一座城市出发有多条未经搜索的边，总会选择对应最小子树（即子树上的城市数目最少）的一条。

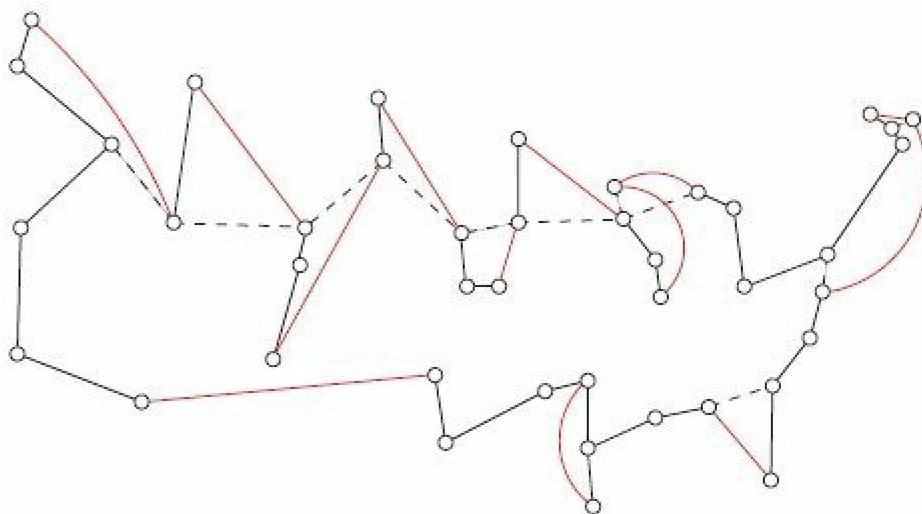


图**4-10**　基于最优树构造路线

4.2.5 Christofides算法

为了给旅行商指路，“种”一棵树是个不错的主意。若想充分发挥树的作用，需要退一步，从莱昂哈德·欧拉的角度思考问题。把一棵树的每条边复制一遍可以生成一个图，按照深度优先搜索算法遍历树的路线实际上相当于周游对应的图的欧拉回路。复制的目的是保证图的每个顶

点都连接偶数条边，不会像哥尼斯堡七桥问题那样出现奇点。[1]

1. 无向图存在欧拉回路的充要条件是每个点都是偶点且该图是连通图。奇点（odd vertex）就是连接奇数条边的顶点。——译者注

我们也可以不复制整棵树，而是对树添加一组边，使每个奇点连接的边数都增加一条。这样一来，生成的图中便没有奇点，因此必然存在欧拉回路，可以截短取直，得到一条周游路线。下面以环游美国问题为例说明这种思路。如图4-11所示，这道题目的树总共包含42个顶点，左图用红色标出了24个奇点，右图中共有12条边染成红色，恰好连接这些奇点各一次。[2]这样的一组边称为一个完美匹配（perfect matching）。Jack Edmonds提出了一种解法，可以在多项式时间内计算出最小代价完美匹配。在最优化领域，这一结论具有里程碑地位，相关内容将在第6章具体讨论。现在，我们暂时只需要知道，因为最优匹配的代价至多只有最优路线的一半（原因稍后阐明），所以最小代价完美匹配就是最合适、最完美的匹配。把树和其最优匹配相结合，在新的图中截出一条欧拉回路，便得到了一条TSP路线，其长度不超过最优路线的1.5倍。在理论上，这样的性能保证已经很不错，而实际使用时，往往可以得到更好的结果。对于环游美国问题，分步求解过程如图4-13所示，最终路线长度为759个单位长度。

2. 原文误作26个奇点和13条边。——译者注

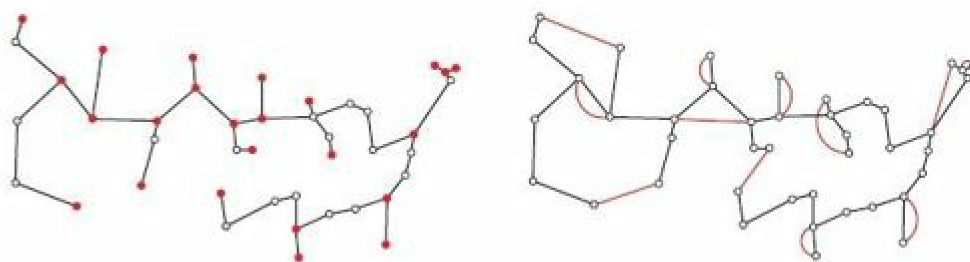


图**4-11**　对于奇点的最小代价完美匹配



图**4-12　Nicos Christofides**，**1976**年

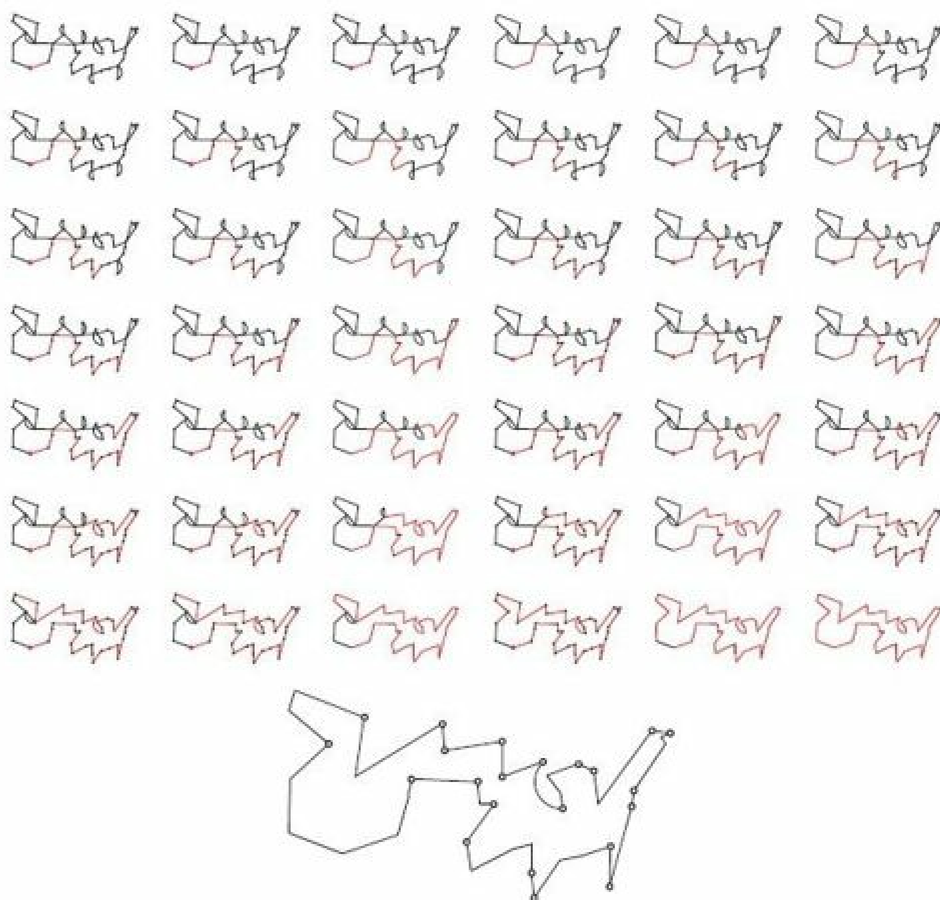


图**4-13**　基于**Christofides**算法构造路线

为了从整体上估算最优路线的长度，首先要注意，在图上沿TSP路

线前进时，会从一个奇点到达另一个奇点，间或穿插几个偶点。如果抄近道，只走奇点，则得到一条周游所有奇点的回路。这条回路可以分拆成两个完美匹配，从某条边开始的第1、3、5…条边组成了一个完美匹配，第2、4、6…条边也组成一个完美匹配。这两个完美匹配中，必有一条的长度不超过奇点回路长度的一半，而Edmonds的最优匹配只可能更短，不可能更长。大功告成！这三步分析如图4-14所示。环游美国的最优路线先被截短，得到一条周游奇点的回路，然后拆分形成两组完美匹配。

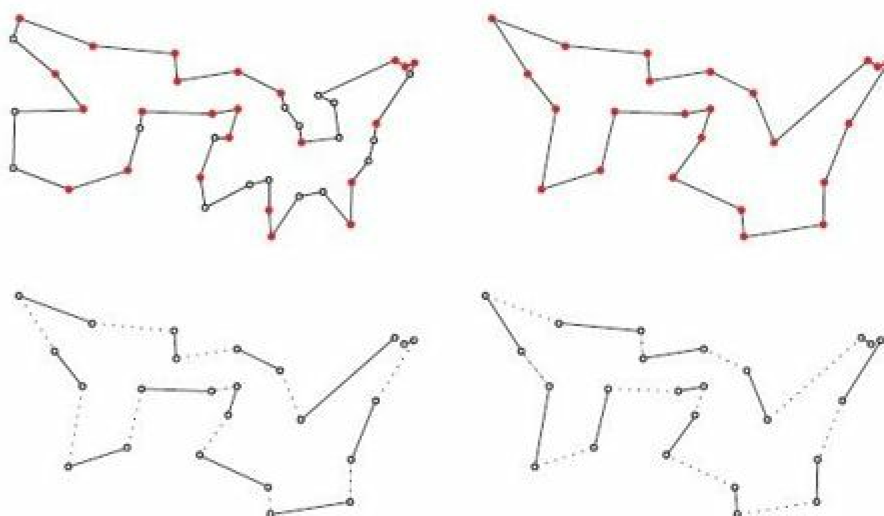


图**4-14**　从周游**42**座城市的最优路线得到两组完美匹配

1976年，Nicos Christofides首次完整提出这一算法，将欧拉和Edmonds的结论结合在一起。Christofides算法在TSP的神殿里占有重要地位，因为已知的其他多项式时间算法都无法保证给出更短的路线上限。[3]

3. Christofides, N. 1976. Report 388, GSIA, Carnegie Mellon University.

4.2.6 新思路

从零开始逐段构造路线的过程简单而纯粹，因此常常吸引新人参与TSP研究，激发新的解题思路，也非常适合亲身体验TSP的复杂性。

如果你有意求解这道题，那么显然有一个目标：改进Christofides算法的性能保证。然而，我有必要警告你，Christofides算法保证路线长度不超过最优解的1.5倍，你可能很难突破这一结果，原因详见第9章。另一方面，如果在实际应用中，新方法能够与已有的路线构造算法一较高

下，也是不足为奇的。除了已经讨论的几种著名算法以外，TSP爱好者和研究者也提出了大量其他方法，包括聚类技术、划分算法、空间填充曲线，等等。目前，在实际计算领域，上述路线构造算法的效果都不如下一节介绍的路线改进方法，但是新的思路完全有可能缩小两者之间的差距。

4.3 改进路线？立等可取！

1985年4月，美国《发现》杂志刊登了一篇介绍TSP的文章，作者是马丁·加德纳（Martin Gardner），插图漂亮地画出了环游美国的路线，如图4-15所示。《发现》杂志读者众多，马丁·加德纳也是家喻户晓的解题高手，强强联手推出的文章自然使旅行商广受关注，但也在读者中引起了骚动。仔细观察图4-15，你会发现骚动的原因——这条周游所有城市的路线显然有好几处可以缩短！

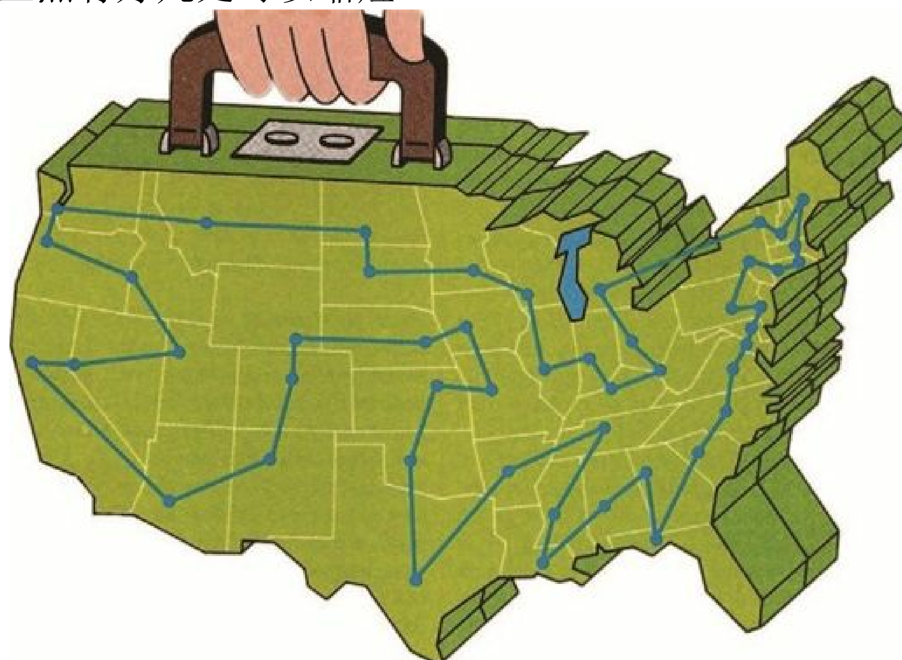


图**4-15**　环游美国的路线，**Nina Wallace**绘制
《发现》杂志，1985年4月，第87页

文章刊出之后，加德纳与IBM公司的数学家Ellis Johnson通过电话。在电话中，他解释说，路线的来源其实是Dantzig等人的论文。文章的问题并不在于路线本身，而是因为编辑画蛇添足，自作主张把各城市挪到了各州首府的位置。那幅插图的说明文字是这样的："旅行商问题是数学界最古老的未解之谜之一。这里是旅行商周游48个州的首府的最短路线。"加德纳真倒霉。当初在1954年，Dantzig等人只是为了方便而选择了那些城市，结果加德纳却因此必须赶紧做出更正，所以他才给Johnson打了电话，Johnson介绍他向TSP专家Manfred Padberg求助。

Padberg当然有能力解决这道周游48州首府的问题，然而当时似乎无法联系到他本人。最终出手相助的是贝尔实验室的Shen Lin。在Gardner

的文章发表4个月后，Lin提出了一条新的路线，也发表在《发现》杂志上。他虽然没有准确求解路线的算法，却是路线改进方法的专家。

这一回，杂志描述路线时非常谨慎。“他的答案对吗？Lin本人对此确信无疑，甚至提供了100美元的奖金，悬赏征求短于10 628英里（约17 104千米）的旅行商路线，要求周游各州首府并使用他给出的数据计算。”编辑向任何有意接受这一挑战的读者提供了一张距离数据表，不过Lin不用担心破财，因为他的路线确实是最优路线。



图**4-16**　环游美国的最优路线，**Ron Barrett**绘制

《发现》杂志，1985年7月，第16页

4.3.1 边交换算法

Lin倡导使用路线改进算法。这类算法名副其实，读取一条路线后，便会查漏补缺，对其调整修正。以《发现》杂志第一篇文章中的TSP路线为例，它在进出田纳西州时形成了一个尖角，表明这部分路线有问题。局部调整修正方法如图4-18所示。首先删掉形成尖角的两条边，同时删掉它们正北方（图中正上方）紧邻的一条边，将路线分为3段，其中一段就是田纳西州首府这一个孤立点。然后再连接3条新的边，将路线连为一体。新添的边用红色表示。由于三条新边的长度之和远远小于3条旧边，因此路线得到了改进。这步操作相当于交换了3条边，因此称

为**3-**交换（3-opt）。



图**4-17**　**Shen Lin**，**1985**年（照片由David Johnson提供）

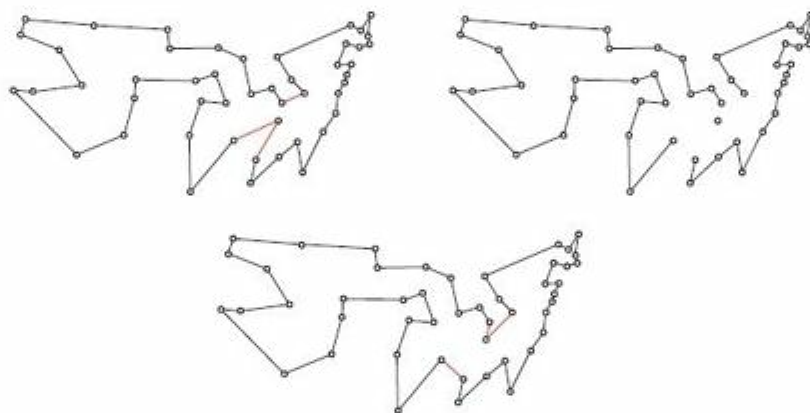


图**4-18**　通过一步**3-**交换改进《发现》杂志的**TSP**路线

Lin在计算这道题目的过程中，全面探寻了可以改进路线的方法，包括2-交换、3-交换和更多边的交换操作。**2-**交换的步骤就是删除路线中的两条边，用另外两条更短的边重新连接路线。首次尝试求解42座城市的环游美国问题时，我们曾按照最近邻算法构建了一条大大需要改进的路线，现在就以此为例探究Lin运用的解题思想。

TSP领域，最基本的定理可能要数这一条：如果题目数据使用欧几里得距离，那么最优路线必定不会自交。用一步2-交换就可以证明这个事实，因为删除一对相交边、重新连接相应点之后，总路线必然会缩

短。如图4-19所示，一步2-交换使总路线长度减小为982个单位长度，缩短了31个单位长度。这步交换非常明显，而且图上仍然有多处可以进行此类交换。

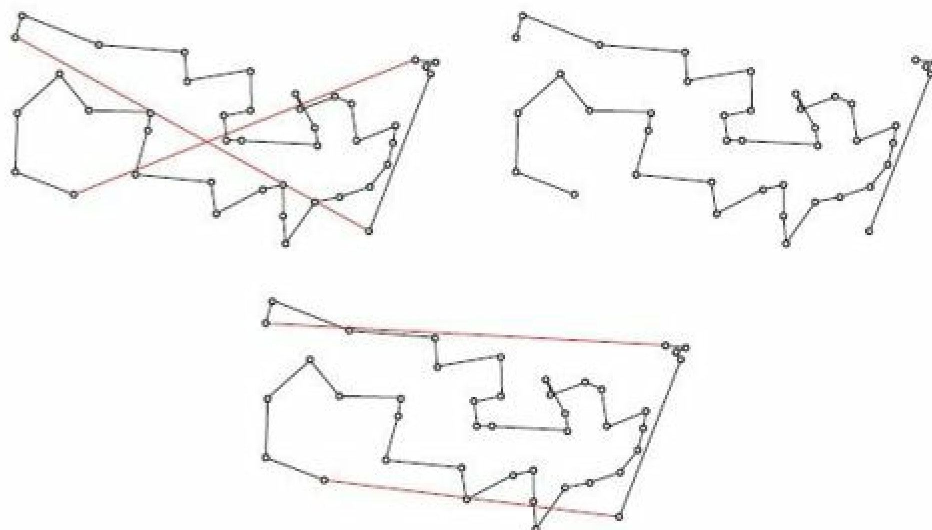


图**4-19**　可以改进最近邻路线的一步**2-**交换

接下来，我们反复使用2-交换改进路线，最终得到的路线如图4-20所示，总长度为758个单位长度。这时总共进行了27次2-交换，已经无法通过这种操作进一步改进路线了。但是我们发现，通过交换两条边的简单操作，就大大改进了差劲的最近邻路线，如今它只比最优路线长8%而已。

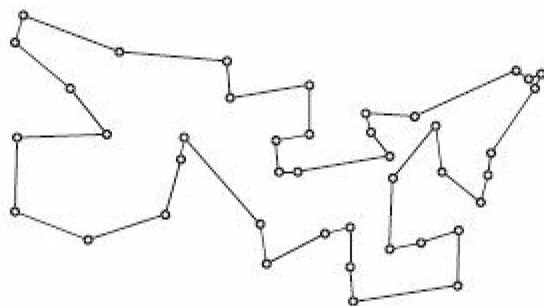
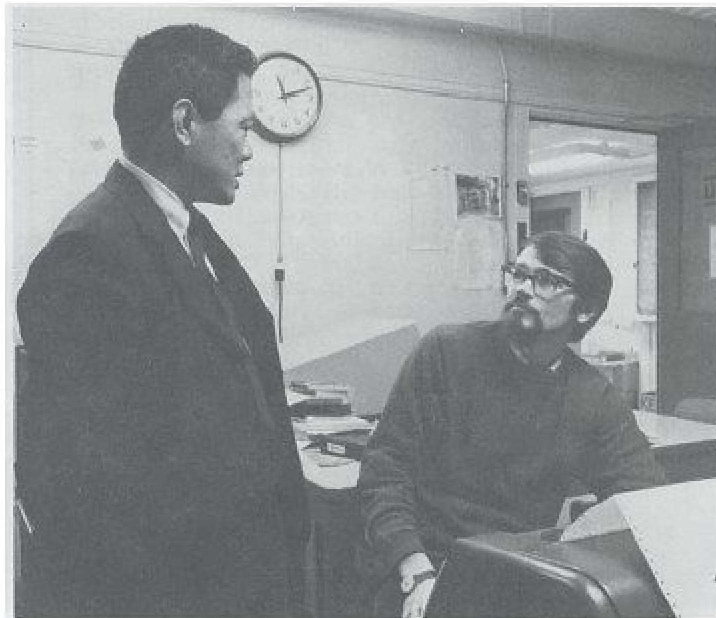


图**4-20**　无法通过**2-**交换进一步改进的路线



“我想我们搞定了。”Shen Lin（左）似乎如此向 Brian Kernighan（右）表示。这两位贝尔实验室的数学家兼计算机专家发明了旅行商问题的一种高效新解法。

图**4-21　Shen Lin**和**Brian Kernighan**，***Bell Labs News***，**1977**年**1**月**3**日（**Brian Kernighan**供图，**Alcatel-Lucent USA Inc.**授权复制）

4.3.2 Lin-Kernighan算法

我们要前进到底，所以接下来可以考虑所有能够改进路线的3-交换，然后是4-交换，再然后是5-交换，以此类推。20世纪60年代中期，Lin确实发表了结果，证明3-交换可以成功地改进路线。但是，如果*k*远远大于2和3，搜寻改进路线的*k*-交换就需要巨大的计算量，因此完全不可行。尽管如此，Lin和计算机科学先驱Brian Kernighan还是成功实现了这种方法。[1]他们巧妙地构建了一种新算法，取得了TSP研究中最伟大的成就之一。

1. Lin and Kernighan (1973).

Lin-Kernighan算法（简称LK算法）精巧复杂，但图4-22足以概述其中心思想。该图将初始路线表现为一个环，降低了理解算法流程的难度。需要提醒读者注意，各边在图中的长度并不代表实际路程。

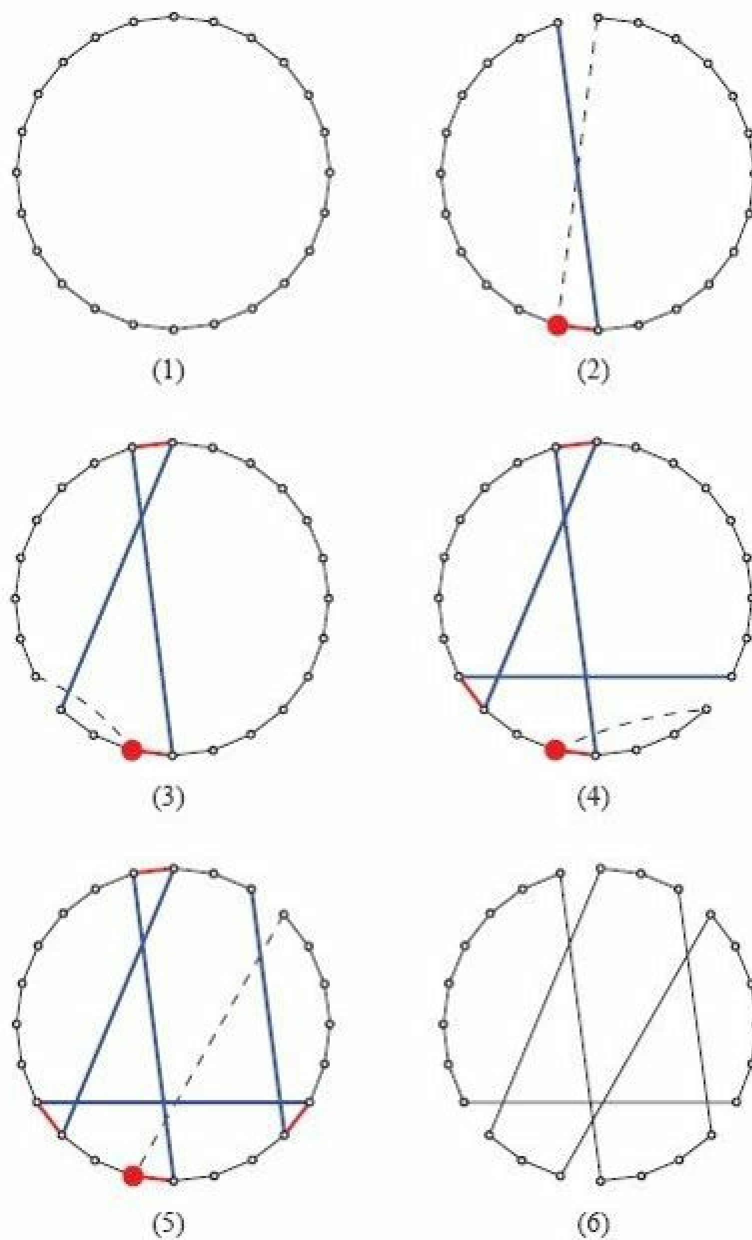


图**4-22**　使用**Lin-Kernighan**搜索算法寻找***k*-**交换

搜索的第一步如图4-22的小图（2）所示。首先选定一座城市作为根据地，图中用红点表示；然后选择一条与之相连且属于初始路线的边，图中以红色表示；再从这条边的另一个端点出发，选择一条不属于初始路线的边，图中以蓝色表示。算法意在删减红边，增加蓝边，因此仅考虑蓝边的代价低于红边的情形。这一步也可以进行图示的2-交换操作，删除路线中与蓝边远端相连的合适的边，并连接该端点与根据地之间的边，从而用蓝边替换红边，即改变路线。如果这步2-交换可以改进路线，那么很好，我们就把具体改进数值记录下来。但是接下来还要继续

搜寻，希望能找到更大的改进。

如图4-22的小图（3）所示，第二步是把刚才考虑删除的边染成红色，并选取不同于刚才考虑连接的边的另一条蓝边。此时，依然仅考虑两条蓝边的代价之和低于两条红边的情形。同样，也可以删除路线中与蓝边远端相连的一条边，增加直接回到根据地的边，如虚线所示。同样，如果这步可能的3-交换可以对路线做出迄今最好的改进，那么我们就把改进的数值记录下来。

搜索继续以蓝边总代价低于红边总代价为前提，考察成对的红边和蓝边。如果到达路线终点，无法增加新的一对蓝边和红边，就回溯到早前的某级端点，继续探寻其他可选的蓝边。该算法最终有两种可能的结局：要么由于时间因素而中止搜索，要么由于已经考虑过所有可能的边而终止搜索。

总之，在搜索结束时，我们选取已记录的改进最大的交换操作，以此改进路线，再对改进后的路线重新开始新一轮搜索。如果找不到有改进的交换操作，则重新选择根据地，再次从初始路线出发尝试搜索。

只需每次选择红边和蓝边，再检查直接回到根据地的边，如此重复即可。乍一听易如反掌，然而细节里暗藏无数玄机。幸好，Lin和Kernighan不仅计算出了出色的结果，还深入浅出地阐述了用于实现与改进该算法的诸多思想。过去40年间，在他们两人的原始论文的指导下，Lin-Kernighan算法得到了精确的设计和落实。新近开发的此类软件能力更强，能够解决超过千万座城市的大规模TSP题目，求出非常优秀的路线。

对于环游美国问题的数据集，以图4-20中经过反复2-交换得到的路线为初始路线，使用Lin-Kernighan算法的解题步骤如图4-23所示。该算法在五次迭代之后找出了最优解，每一步所得路线中红边所示的路段都将在下一轮搜索中被删除。

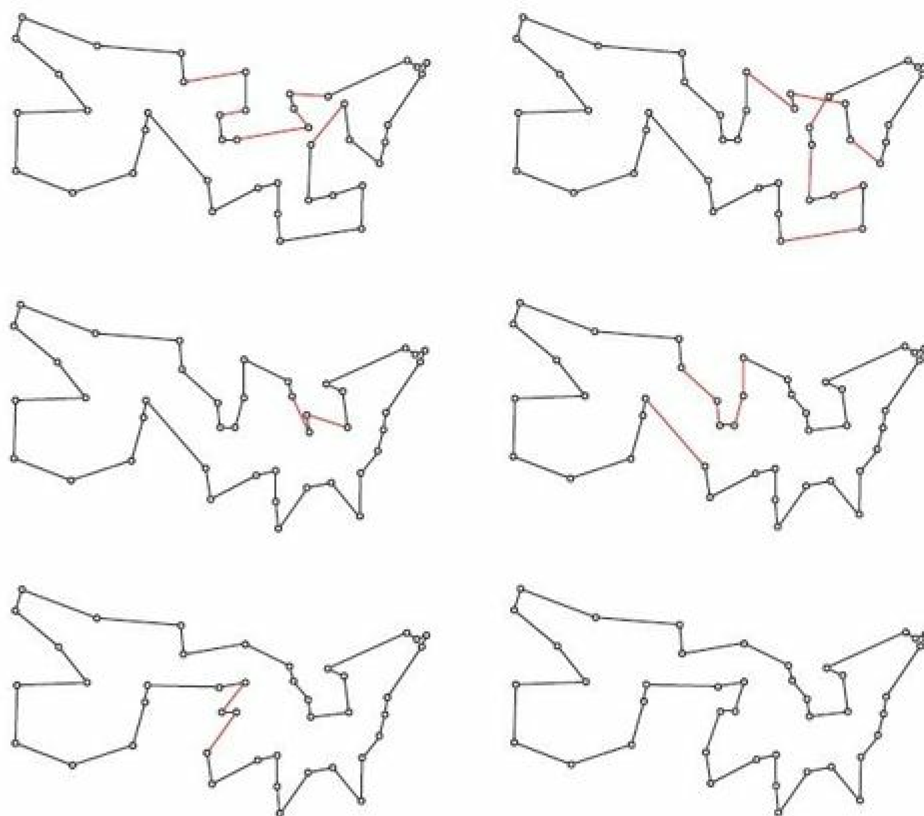


图**4-23　Lin-Kernighan**算法求解环游美国问题的五次迭代

Lin和Kernighan在原始程序中使用随机路线作为初始路线，结果同样得到了环游美国问题的短路线，这本来就在意料之中。“对于中小规模的题目，比如环游42座城市问题这么大的规模，一次试验得到最优解的概率接近1。”[2]然而后来的结果却出人意料，卓越非凡。他们设计这一基本方法原本只为解决几十或几百座城市的TSP题目，却为过去几十年中大多数优秀TSP启发式解法奠定了基础，在此期间，规模大得多的题目也获得了解决。

2. Lin and Kernighan (1973).

这里必须指出，*k*-交换方法虽然在应用中表现出了优秀的性能，在理论上却无法保证优良的最差情况性能，实属不幸。以反复进行2-交换的方法为例，对于满足三角不等式的题目来说，该方法只能确保路线长度不超过最优解的$4\sqrt{n}$倍。[3]这是Lin-Kernighan算法的缺点所在。即便如此，使用该算法时也不必过分担心，因为它是算法中的王者，一般情况下都确实能够得到非常好的解。

3. Chandra, B., H. Karloff, C. Tovey. 1994. In: *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 150–159.

4.3.3 Lin-Kernighan-Helsgaun算法

Lin-Kernighan算法在实际计算领域长期占据统治地位，得力于研究界对它持续不断的改进和增强。尽管多数改动只是在原有想法的基础上进行细微的调整，但是1998年，计算机科学家Keld Helsgaun却带来了一枚重磅炸弹。

Helsgaun的主要贡献是重写了算法的核心搜索程序。此前25年间，这部分内容几乎一直原封未动。Lin-Kernighan标准算法可以视为搜寻2-交换操作序列的方法，Helsgaun的新方法却以5-交换操作序列为搜寻对象。换言之，他没有沿用一条红边一条蓝边的逐步搜索模式，而是发明了一种新的方案，每次同时考虑5条红边和5条蓝边，总共10条边。

10条边。看到这个，你的第一反应想必是："边数可真够多的。"没错，如果考察5条红边、5条蓝边的所有可能情形，确实会严重拖慢算法速度。为避免这种结果，Helsgaun对搜索加以限制，只考察能由连续5步搜索得到的红边、蓝边组合，该过程中并未要求每步的蓝边代价之和必须低于红边。他的方法每次都一下子考虑所有此类5-交换，因此能够发现原有标准算法完全找不到的改进操作。

新算法名为Lin-Kernighan-Helsgaun算法，简称为LKH算法。它以5-交换操作为核心，还借用了大量各类技巧，树立了探寻路线方法的新标杆。"对于含有100座城市的一般题目，给出最优解只需不到一秒钟；若含有1000座城市，则只需不到一分钟。"[1]当时人们认为该领域的研究已经相当成熟完善，但LKH算法还是在实际性能方面实现了如此惊人的飞跃。

1. Helsgaun, K. 2000. Eur. J. Oper. Res. 126, 106–130.

4.3.4 翻煎饼、比尔·盖茨和大步搜索的LKH算法

人们得知Helsgaun构建了许多知名难题的改进路线后，纷纷猜测他在实际计算中是如何成功实现5-交换策略的。为使读者理解其原因，在此有必要指出，在Lin和Kernighan发表原始论文之后、LKH算法公布之前的25年时间里，实现标准LK算法的有效程序代码屈指可数。LK搜索算法虽然有清晰明确的解释表述，但难以转化为能处理大规模数据集的软件。

LK算法还可以用难来形容，但LKH算法看起来简直是不可能实现

的。事实上，LK算法按照红边、蓝边交替顺序搜寻路线，具有一大特色，即在搜寻的每一步，回到根据地城市的路线都仅有一条。换句话说，如果从一条完整路线中删除两条边形成两条子路线，那么把它们连接起来形成新路线的方式是唯一确定的。然而LKH算法却不同，简单计算可知，每一步5-交换操作都来自连续五步2-交换，涉及5条子路线，连接形成新路线的可能方式便有148种之多。所以LK算法和LKH算法之间的差别，就是1和148之间的差别，也就是难和特别特别难之间的差别。

Helsgaun向研究界公开了完整的程序代码，以此揭示了他取得成功的秘诀。Dave Applegate和我通读了他的文件，结果发现其中根本没有真正的独门秘诀可言：他的代码里完整列出了148种情形，分别讨论了所有的可能性。他为了写出正确可行的代码，实现极为复杂的算法，付出了堪比愚公移山的努力。

Helsgaun的代码令人动容，LKH算法的性能振奋人心，但人们还是好奇，把5-交换改为6-交换能否进一步优化结果。Applegate写了一个小程序，计算出连续得到的6-交换会产生1358种重连路线的可能方式。这样的数字已足以令人望而却步，但每次交换的边数更大会怎样呢？升级到9-交换的时候，要面对的可能情形已高达2 998 656种，确实会相当费劲。

不过，其实还有一线希望。Applegate的小程序能够逐一列出所有可能的重连方式，而仔细考察LKH算法可以发现重连方式的规律。因此，我们结合这两方面内容，能够设计出一个计算机程序。对于任意k，这个程序都能提供解决k-交换操作的实际代码，也就是说用一个程序生成另一个程序。

这个主意似乎不错，实际上产生的代码非常长：6-交换的代码有120 228行，7-交换的代码有1 259 863行，8-交换的代码有17 919 296行，全都是用C语言编写的。尽管把这些代码编译为计算机可以识别的2进制语言并非易事，不过生成的程序确实可以运行，而且得到了有趣的结果。8-交换是能够实现的上限，效果却并不比5-交换更让人满意。而对于9-交换而言，根本不可能生成完整列表逐一讨论所有情形。

Applegate毫不气馁，勇往直前。他想到，如果采用这种母程序生成子程序的方法，提高母程序的工作效率，就不必逐一列举出所有情形。具体说来，母程序可以在执行k-交换搜索的同时，提供每种情形的应对步骤，从而生成子程序代码。这样一来，虽然执行搜索步骤所需的计算时间仍然是限制因素，但是有机会实现每次交换更多边数。加快生成程序计算速度的研究与翻煎饼问题[1]密不可分。微软公司的比尔·盖茨

（William Gates，常称为Bill Gates）本科就读于哈佛大学，他在校时曾和TSP专家Christos Papadimitriou共同研究过这一类问题，此事众所周知。翻转煎饼堆顶部的一叠煎饼对应于反转TSP路线中的一段子路线，也就是2-交换操作。因此，要想实现能够生成*k*-交换程序的母程序，首先要有一个算法，能够找出按照*k*-交换顺序重排路线最少需要反转几次。这正是Gates-Papadimitriou的研究成果的变形。[2]我们经过努力实现了这种算法，得到了连续*k*-交换的高效动态搜索机制。

1. 翻煎饼问题是指，有一堆大小不一的煎饼叠放在一起，希望通过翻转把煎饼按照从小到大的顺序从上到下排列，要求编写翻转的算法或程序。每次翻转时，把铲子插入两块煎饼之间，铲子上方的煎饼整体上下翻转，使铲子上最下层的煎饼翻到顶上，而原来顶上的煎饼则落到插入铲子的地方，铲子下方的煎饼没有变化。参见http://poj.org/problem?id=2275。——译者注

2. Gates, W. H., C. H. Papadimitriou. 1979. Discrete Math. 27, 47–57.

Helsgaun后来对LKH程序进行了有力的升级，在新的代码中引入了类似的思想，允许用户具体指定每次交换操作包含连续几步。2003年，为了展示新软件的效力，他以周游瑞典24 978座城市的问题为例，计算了10-交换操作，所得TSP路线的最优性于次年得到证明。

4.4 借鉴物理和生物思想

事实证明，把TSP视为一类搜索问题的特例，从全局着眼看待寻找路线过程，这种出发点无论对于找到好的路线还是想出多用途的技术都大有裨益。思路意图在于产生元启发式算法（metaheuristic），即用来设计启发式算法的启发式算法。这类研究工作本质具有一般性，因此科学界各领域的研究者纷纷前来参与搜寻路线之旅。

4.4.1 局部搜索与爬山算法

该研究领域中有一个类比很有价值：想象路线所处环境具有一定的地形变化，每条路线的高程对应于它的好坏程度。脑海中的图像恰似加舒尔布鲁木诸峰，如图4-24所示，好的路线对应于各座山头，而最好的路线位于高耸入云的加舒尔布鲁木II峰之巅。这样一来，启发式算法就可以想象成四处移动搜寻高地的过程。



图**4-24**　加舒尔布鲁木诸峰（**Florian Ederer**供图）

为使这种想象图景有意义，应当对两条路线在什么情况下彼此相邻加以说明。通常的处理方法是在每条路线周围设定邻域（neighborhood）。例如，可以把两条路线相邻定义为它们能够通过一步2-交换操作互相转化，或者定义为能够通过LK算法找到的某种交换操作互相转化。大的邻域有助于在地理环境中确定前进方向，而构建邻域

时应当使算法能查看并评估近邻。

爬山算法（hill-climbing algorithm）代指一类路线改进方法，以反复操作改进性的2-交换为例。这类算法之所以得名，是由于其过程可视为沿一系列相邻路线前进，始终向高处移动。算法的每一步都进行一次局部搜索（local search），以寻找附近的高地。若搜索全面彻底，则最终算法将在山峰上终止，或者至少在高原上终止，因为此时所有局部移动必然要么走下坡路，要么走平路。完整运行时，算法会从初始路线出发，然后迅猛攀升到局部高峰处。

请注意，初始路线的选取将决定爬山算法的命运：如果初始路线只能到达小山坡的半山腰处，那么算法就会受到局限，最终路线将会表现平平，只能登顶那座小山而已。 因此，Lin和Kernighan决定选择随机路线作为反复运行LK算法的初始路线。理由何在？想象向所处地面投掷飞镖，若飞镖数量足够多，便有很大概率击中高峰周围的山坡。

随机路线可以使飞镖落点广泛分散开，但是也有缺点，就是它们通常不是好路线，所以初始阶段往往会在低洼的山谷中蜿蜒很久。如果TSP题目的规模很大，从山谷走到山峰就可能需要很长时间。也可以选择折中的方法，通过随机选择出发城市保证路线的随机性，再按照最近邻原则投掷飞镖。

4.4.2 模拟退火算法

模拟退火（simulated annealing）算法是一种启发式算法。它按照一定概率接受比当前解更差的邻域，因此条件限制比爬山算法更弱。这种情况下的接受概率一开始比较高，然后随着算法运行而逐渐降低，使算法有机会先跳到更高的山坡上，然后才转而稳步向上攀登。

模拟退火算法由Scott Kirkpatrick、Daniel Gelatt和Mario Vecchi发明。他们的原始论文研究了TSP，并针对一道400座城市的题目公布了启发式算法得到的若干路线。[1]他们写道，该方法的动力源于统计力学与TSP的关联，在统计力学中，退火表示先加热再缓慢冷却材料以改善其结构的工艺过程。

1. Kirkpatrick, G., C. D. Gelatt, M. P. Vecchi. 1983. Science 220, 671–680.

迄今为止，模拟退火算法尚未给旅行商带来多少好处，但它作为通用搜索方法却获得了盛大的成功。谷歌学术搜索数据表明，原始论文引用次数超过18 700次，这样的引用率几乎是前所未有的。

4.4.3 链式局部最优化

模拟退火算法给当前的寻找路线方法带来的最大影响很可能不在于这种方法本身，而在于它向TSP研究领域引入了物理学的思想。事实上，计算结果之所以能够首次突破反复LK算法所能达到的界限，应当归功于物理学家的跨界贡献。

20世纪80年代末，加州理工学院物理系的Olivier Martin、Steve Otto和Edward Felten三人提出了不同于“投掷飞镖”随机策略的另一种方法。他们的想法是，像LK算法这样强大的局部搜索算法得到的路线往往会延伸到所处区域内的高地，因此可以对这一点加以利用。于是，Martin等人提议，算法第二次运行时不再从随机城市出发，而是首先观察当前所处的山峰周围，判断能否越过附近的几处壁垒，到达通向更好地点的另外一座山坡上。

具体的方案是通过变换原有的LK路线获得一条新的初始路线，而不是随机投掷飞镖确定。算法全过程多次重复这一变换步骤，一发现更好的路线就用它替换原有路线。要想使算法有效运作，路线在变换后必须离开原有邻域，因此这种改动难以通过LK算法还原。Martin等人发现，如图4-25所示类型的随机4-交换就可以圆满完成任务。

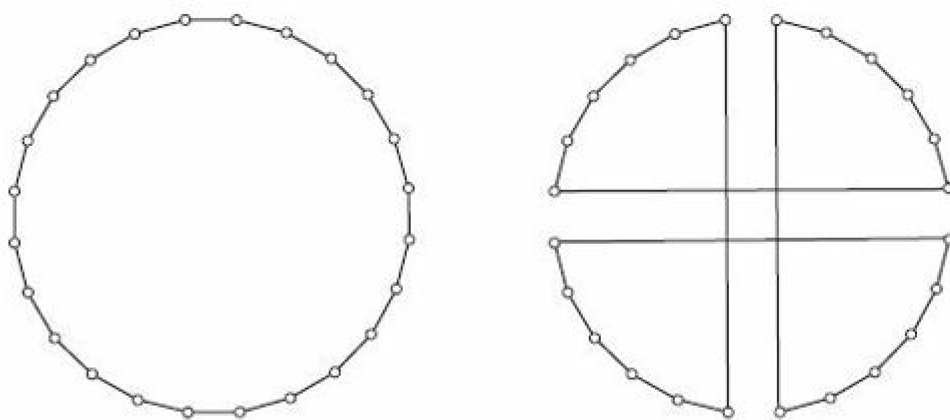


图**4-25**　双桥变换

这样得到的算法性能出色，称为链式Lin-Kernighan算法（Chained Lin-Kernighan），即链式LK算法。这种思想之所以成功，得益于种种机缘巧合。首先，Martin等人的直觉是正确的，通过kick机制访问邻近区域确实可以更好地认识周围山峰的情况，LK算法本身则用来引领路线在山坡上爬升；其次，路线经过变换后，运行LK算法的速度确实比随机路线快得多，原因很简单，毕竟变换后的路线大体形状比较好，所

以算法无需多次迭代即可获得局部最优解。

链式LK算法的程序实现基本上垄断了20世纪90年代的路线搜寻领域。Concorde程序就用到了该算法的一个版本，对于城市至多可达100 000的题目，通常也只需一两秒便可找到超出最优解不到1%的路线。若想获得更好的解，可以选择使用LKH算法。然而如果题目的数据集非常大，那么链式LK算法依然占据统治地位。例如，有一道题目包含25 000 000座城市，各点落在25 000 000×25 000 000的正方形内，横纵坐标均为整数，路线数据取为欧几里得距离，计算结果如图4-26的曲线所示。2000年的计算机在8天时间内找到了一条路线，其长度大约比最优路线长0.3%。[1]

1. Applegate, D., W. Cook, A. Rohe. 2003. INFORMS J. Comput. 15, 82–92.

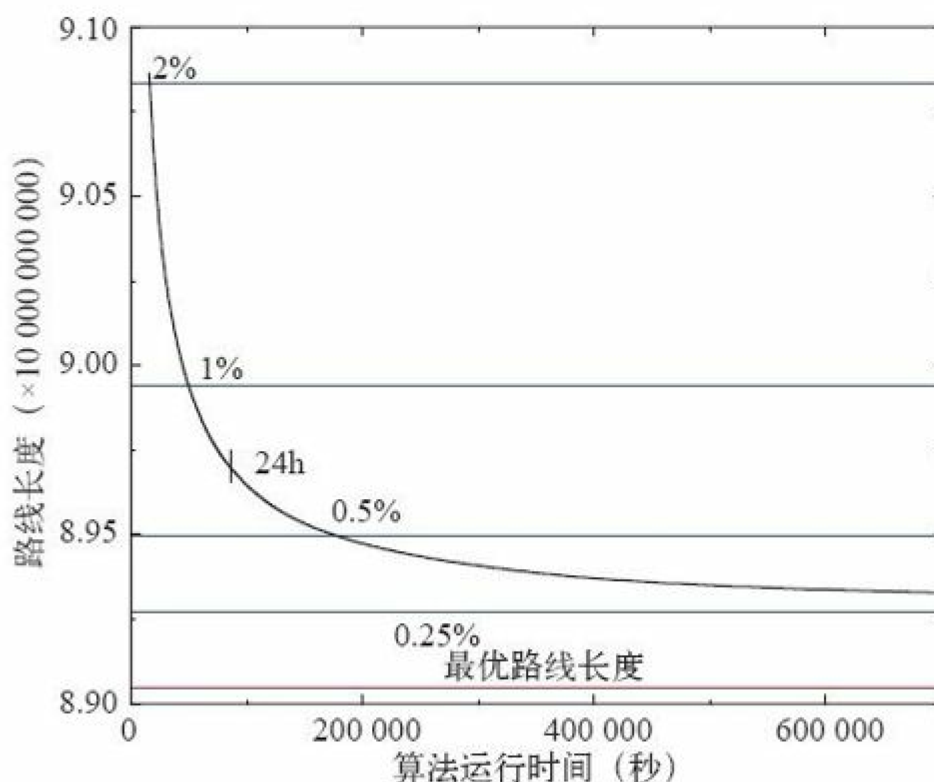


图**4-26**　应用链式**LK**算法求解一道题目的情况，该题包含**25 000 000**座城市，数据取欧几里得距离

4.4.4 遗传算法

有人如上所述，以地理思路看待旅行商路线，也有人从生物角度出

发，把路线当成能够逐渐变异和进化的生命有机体。有一类算法应用了后一种思维方式，称为遗传算法（genetic algorithm），其灵感来源是John Holland出版于1975年的里程碑式著作*Adaptation in Natural and Artificial Systems*（《自然系统与人工系统中的适应性》）。[1]Holland并未讨论如何解决TSP，但不久后，他的遗传算法思想便出现在寻找路线的文献中。

1. Hollan, J. 1975. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, Michigan.

在TSP领域，遗传算法的思路大致如下。首先形成路线的初始“种群”，可以通过反复从随机城市出发使用最近邻算法获取路线。然后是一般的循环步骤，在种群中选取若干对路线，让它们“交配”（即交叉配对）得到子代路线[2]，再从父子两代路线中选出一个新的种群。反复多次进行这一步骤，最优秀的路线最终将脱颖而出，成为唯一的生存赢家。

2. 针对其他类型问题的遗传算法时常还有一步“变异”步骤，即种群的个体发生突变。

遗传算法的实质是模拟自然界的进化过程。类比饶有趣味，但请牢记在心，仅仅借用达尔文的语言并不意味着最终一定能得到好的路线。事实上，早期的TSP遗传算法并无可圈可点之处，即便题目规模限制得非常小，结果也不太出色。然而，维持路线的种群这一想法具有明显的优势。根据一般性的遗传算法，可以精心设计出威力强大的启发式算法，结合局部搜索方法使用则效果更佳。

在遗传算法的基本原则框架下，路线种群进化的方法各式各样，为我们留有充足的选择空间。我们不但可以选择交叉配对过程，也可以选择适应度评价标准（fitness measure）以确定如何选取下一代种群。关于这一评价标准，人们提出了一些绝妙的思路，力求在解的品质和种群的差异性之间达到最佳平衡。

就交叉配对过程本身而言，早期的遗传算法方案试图在一条父代路线中搜寻能够被另一条父代路线的子路线替换的部分子路线。这种做法限制性很强，题目规模较大时尤甚。另外一种做法则更有效，即对于两条父代路线A和B，在A中选择一段子路线，然后尽量选择B和A的边（优先选择B的边），把它扩充为完整路线。还有一种配对方法如图4-27所示，称为边交叉（edge-assembly crossover，简称EAX）算法。图中有两条环游美国的路线，分别染成蓝色和红色。为了将两者组合到一起，我们取它们包含的边的并集，并在相应的图中选取一条各边颜色红蓝交替的回路，如图4-27中小图(3)所示。然后，我们考虑原来的蓝色路线，删去出现在新回路中的蓝边，再添加上新回路中的红边，如小图(4)

所示。最后，我们通过一步2-交换操作，把上述过程中产生的子路线合并为一条完整路线，如小图(5)和(6)所示。

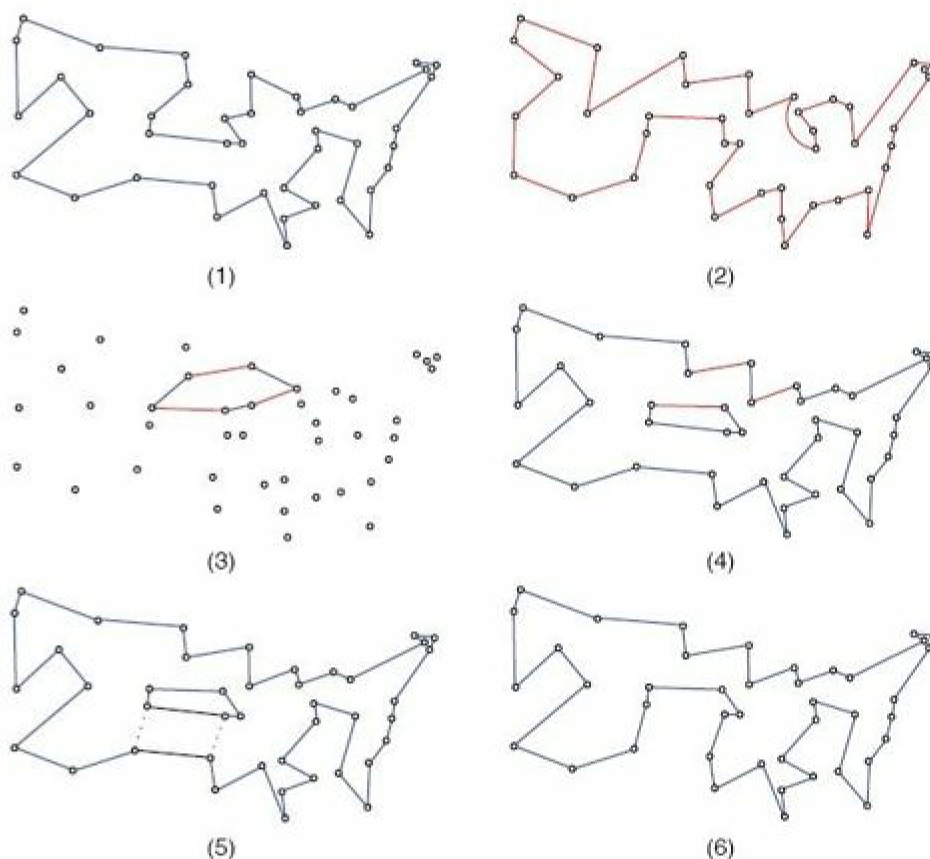


图**4-27**　两条路线交叉配对

永田裕一采用EAX配对方案，发明了一种卓有成效的寻找路线方法，时至今日仍鲜有敌手。[3]该算法的实现以EAX的超高速实现为基础，可以连续处理多代路线。他成绩斐然，比如他为100 000座城市的《蒙娜丽莎》一笔画TSP找到的路线就是目前已知最好的一条。

3. Nagata, Y. 2006. Lect. Notes Comput. Sci. 4193, 372–381.

4.4.5 蚁群算法

也许读者曾经遇到过这样的情景：自己的食物不幸落入一群饥肠辘辘的蚂蚁的口中。这群讨厌的小虫通常排成一条长队，浩浩荡荡向你的花园甚至室内进军。一只蚂蚁单独行动时并无章法可言，但是整个蚁群通过信息素轨迹相互交流，却能成功找到省时省力的路线。受到蚂蚁集体行为的启发，一类TSP启发式算法诞生了，这就是蚁群优化算法

（ant-colony optimization，简称ACO）。

蚁群算法研究的领军人物是比利时人Marco Dorigo。1992年，他在博士毕业论文里提出了该算法的思想。[1]此类算法的工作原理是令一小群蚂蚁沿着图的各边移动，每只蚂蚁探寻一条路线，每次到达新顶点时都选择一条通向未访问过的顶点的边。选择规则是算法的核心，选择的依据是每条边对应的信息素值（pheromone value），信息素值高则选中概率也高。等到所有蚂蚁都走完路线后，各边的信息素值会根据一定的规则变化，增量与路线长度成正比，因此好路线各边的信息素值增加程度大于坏路线的边。

1. http://www.aco-metaheuristic.org.



图**4-28**　求解**TSP**的蚂蚁（**Günter Wallner**供图，原图出自**Georg Glaeser**和**Konrad Polthier**的著作***Bilder der Mathematik***一书）

这种方法自然直观，令人心动，然而它至今未表现出与改进的LK算法一较高下的能力。不过，最近几年间，ACO算法在其他领域大显身手，有效解决了诸如时序安排问题、图着色问题、分类问题和蛋白质折叠问题等形形色色的难题。这个活跃的研究题目有力地佐证了对旅行商

的关注如何能给研究者指明方向，使他们发现有趣的通用方法，可以求解具有各式各样的应用背景的最优化问题。

4.4.6 其他

TSP的各种元启发式思想得到了大量运用，本节只述及了其中性能最优秀的几种，还有许多其他方案，如神经网络算法（neural network）、禁忌搜索算法（tabu search）和人工蜂群算法（honey bee）。如果你想到了一种通用的搜索机制，那么在开发、改进、测试及比较这种新方法时，即使你预想的应用领域与小小旅行商的奔波路线相去甚远，TSP也是非常合适的练兵场。

4.5 DIMACS挑战赛

各领域的研究者都在热火朝天地寻找TSP路线，这种涉猎领域的广度虽然是TSP的优势，但是也曾导致人们对当时的技术水平认识不足。事实上，在20世纪80年代，《自然》杂志之类的顶尖科学期刊都曾刊载有关计算的论文，而文中描述的TSP题目仅包含三五十座城市，发表的结果也往往只是弱逼近。然而与此同时，LK算法已经可以保证在转瞬之间给出最优解，Martin Grötschel和Manfred Padberg也在用精确算法求解包含成百上千座城市的题目。

20世纪90年代，两大事件试图解决这种矛盾状况。第一件事是TSP 90大会，举办方是莱斯大学并行计算研究中心。来自世界各地的研究者济济一堂，既有精确算法方面的专家，如Grötschel和Padberg等人，也有研究寻找路线方法的课题组。TSP的测试题目数据库TSPLIB由德国海德堡大学的Gerhard Reinelt创建。它于1991年面世，就是这次大会的重要产物。TSPLIB数据库收集了来自学术界和工业界的逾百道TSP难题，为各地区各学科的研究人员提供了公共的试验台。[1]

1. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

第二件则是DIMACS的"TSP挑战赛"（TSP Challenge），发起人是AT&T研究所的David S. Johnson。DIMACS全称为具体数学与理论计算机科学中心（Center for Discrete Mathematics and Theoretical Computer Science），坐落于罗格斯大学。20世纪90年代，该中心举办了一系列有关算法实现的挑战赛，其中最著名的就是TSP挑战赛。[2]

2. http://www2.research.att.com/~dsj/chtsp/about.html.



图**4-29**　左图：**Martin Grötschel**、**Gerhard Reinelt**和**Manfred Padberg**；右图：**Robert Tarjan**、**Dorothy Johnson**、**Al Aho**和**David Johnson**

> TSP挑战赛的一大目标是纵览TSP启发式算法领域技术发展水平的

> 现状，描绘出可以再现的全景图。这样，以后的算法设计者就能很快自行判断，新算法与现有TSP启发式算法相比有何优劣之处。

DIMACS向全世界的寻找路线研究者发出了征集请求，并收到了130种不同的算法和程序实现，最后得到了两项重大成果：其一是一家网站，研究者可以在网上直接对比不同解法；其二则是一篇优秀的综述，作者是Johnson和另一位组织者Lyle McGeogh。[3]

3. Johnson, D. S., L. A. McGeogh. 2002. In: G. Gutin, A. Punnen, eds. *The Traveling Salesman Problem and Its Variations*. Kluwer, Boston, MA. 369–443.

Johnson在组织挑战赛方面付出了巨大的努力，他个人对于寻找路线的方法也做了详尽的计算研究。他的工作大大推动了算法工程的发展，决定了该领域的现状。2010年，他获得了美国计算机协会颁发的高德纳奖（Knuth Prize），获奖理由是他为算法理论分析与实验分析作出了杰出的贡献。他作为TSP研究的世界级领袖，这样的表彰当之无愧。

4.6 路线之王

启发式算法需要在运行时间和路线质量之间权衡取舍。有些算法成本最为昂贵，人们愿意花费超长的计算时间，只求达到可以实现的最好结果。他们的竞争毫无限制，相互毫不手软，热烈程度堪比一级方程式赛车。选手竞相寻找周游难题数据集的路线，力图突破当前最好路线的长度。

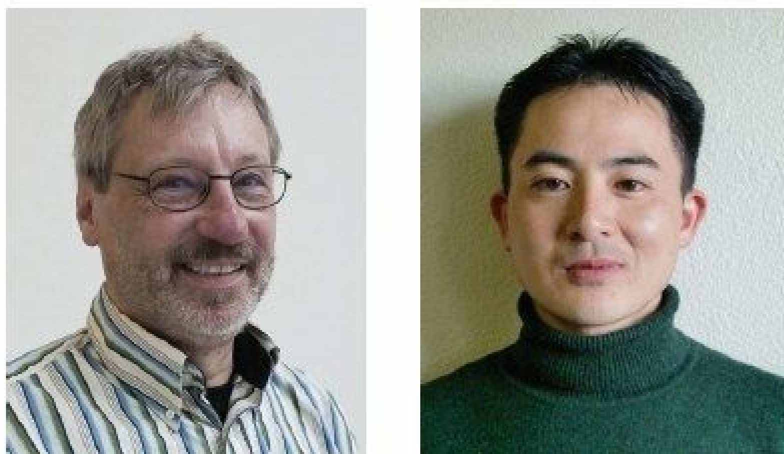


图**4-30　Keld Helsgaun**（左）和永田裕一（右）

毫无疑问，在该领域，丹麦人Keld Helsgaun和日本人永田裕一双双荣登世界冠军的宝座。Helsgaun的LKH程序自从1998年诞生以来，一直是寻找路线领域的标杆，而他也在不断扩展和改进他的算法，向其注入诸多新鲜的思想。对世界旅行商问题，他的路线是迄今最好的；对破纪录的85 900座城市TSP，他的路线是绝对最优的；在VLSI Test Collection的排行榜上，他的名字随处可见。[1]不过永田裕一也不甘落后，他实现了一种TSP遗传算法，并以此求出了《蒙娜丽莎》TSP挑战的已知最佳路线，也给出了National TSP Collection中规模最大的两道题目的最佳答案。[2]你如果有一道大规模的题目，希望得到出色的答案，找这两位就对了。

1. http://www.tsp.gatech.edu/vlsi/index.html.
2. http://www.tsp.gatech.edu/world/countries.html.

第5章 线性规划

> 我认为，线性规划方法的发展，是20世纪的数学对解决工商业界的实际问题所作出的最重大的贡献。

——Martin Grötschel，2006年[1]

1. Grötschel, M. 2006. Notes for a Berlin Mathematical School.

TSP的全部挑战就是既要选出周游给定点集的最优路线，同时还要清楚它确实是最优的。如果使用暴力算法枚举所有排列情形，解题者自然可以确保达到要求，然而这种方法失之直白呆板，又缺乏实际应用的高效性。人们需要的算法应当又能保证路线质量优秀，又无需逐一检验每条可能的路线。在这样的背景下，效率惊人的线性规划（linear programming）便是不二之选。线性规划方法结合所有路线都满足的一系列简单规则，最后提炼出一条规则，形如“所有遍历给定点集的路线长度都不短于X”。X的数值提供了直截了当的路线评估标准：如果我们能构造出一条长度恰为X的路线，那么它必然是最优路线。

听起来像魔法一样神奇，不过事实上，Concorde程序乃至目前所有TSP精确解法确实都采用了线性规划方法。而且，线性规划可以应用于TSP以外的问题，因此它已成为现代数学中最成功的传奇之一。

5.1 通用模型

线性规划的故事有个经典的开头：1939年，年轻的George Dantzig在美国加州大学伯克利分校读研究生一年级。有一天，他上课迟到了，这门课的授课教师是Jerzy Neyman。Dantzig进了教室，看到黑板上写了两道题目，便匆匆忙忙抄下来，几天后把解答交了上去。“长话短说，我把黑板上那两道题目当成作业做完了，可是它们实际上却是统计学领域著名的未解难题。”[1]这一周的辛苦研究真不赖。那两道“作业题”的解答最后成为了他的博士毕业论文的主要内容。

1. Albers and Reid (1986).

Dantzig从伯克利毕业时，正值第二次世界大战。二战期间，他一直供职于美国空军研究规划问题。尽管“规划”的英文是program，与“程序”相同，但是在军事环境中，这个词指的并非一系列计算机指令，而是“针对作战单元的军事训练、后勤供应与调度部署而提出的行动计划”。[2]Dantzig逐渐成为了该领域的专家，谙熟如何用台式计算器处理各种报表系统提供的数字，从而给出此类规划。

2. Dantzig (1991).

战争结束时，美国国防部向Dantzig提供了一个优越的职位，让他继续留在五角大楼。这份工作薪资丰厚，但限定了研究方向。国防部的Dal Hitchcock和Marshall Wood指明要他实现军队规划流程的机械化。Dantzig素来直面挑战，这次也不例外。他迎难而上，发明了一种规划理论。这种理论称为线性规划（linear programming，简称LP），具有深远的意义。

5.1.1 线性规划

20世纪30年代，Wassily Leontief曾经创立一种经济模型，其中指明了生产过程的输入与输出之间的平衡。Dantzig对于线性规划的研究很大程度上受到了这一成果的影响。他提出对经济活动中的选择加以限制，把原来局限于生产平衡的思想拓展为一般理念。



图**5-1　George Dantzig**（**Mukund Thapa**供图）

在Dantzig的线性规划模型中有三大核心要素。第一，他的约束条件不限于相抵的等式，也可以包括不等式（inequality），用来限定一个量至少应该和另一个量一样大。这种新特色常常用来声明某物质的数量至少也应该是零，Dantzig请出疯帽匠帮他解释这一点。[1]

1. Carroll, L. 1865. *Alice's Adventures in Wonderland*. Project Gutenberg Edition.

> 三月兔诚心诚意地对爱丽丝劝道：“再多喝点儿茶吧。”
>
> “我还一点儿都没喝呢，”爱丽丝不高兴地答道，“所以我没法再多喝点儿。”
>
> “你是想说不能再少喝吧，”疯帽匠说，“因为要喝得比‘没有’多是很容易的。”

疯帽匠说得很对。有些物质的量只有取非负数[2]的值才有意义，比如爱丽丝喝下了多少茶，因此，如果用变量T代表她喝的茶的量，就要限制T至少为零，简单记为$T \geq 0$，其中≥是大于等于号，代表前面的数大于或等于后面的数。

2. 请注意“非负数”不同于“正数”，因为正数是严格大于0的数。

线性规划的第二个核心要素就是对约束条件加以限制，要求它们都是线性的。William Safire曾在他的“论语言”（On Language）专栏上阐述他对“线性”一词的看法，“线性思维往往具有贬义，意思等同于‘缺乏想象力’或者‘过分逻辑化’，然而线性规划却力图处理系统各部分之间的持续相互作用。”[3]在Dantzig的理论里，他假定活动消耗的资源正比于活动的强度。如果爱丽丝喝一杯茶要放两块方糖，那么我们就认为她喝两杯茶时会放四块方糖，也就是说，活动强度加倍，则资源也应加倍。本例的表达式为S=2T，即S−2T=0，其中S是糖的量。你会发现S−2T=0是一条直线的方程，“线性”正是因此得名。

3. Safire, W. 1990. “On Language”. *New York Times*, February 11.

模型用一组变量表示不同活动的强度和物质的数量，一般每个约束条件从这组变量中选取若干个相互加减，从而将它们组合在一起。所以如果在一个线性规划模型中有变量A～Z，那么可能有下面的约束条件：

$A+B+C+D \geq 100$

$2E+8G-H=50$

$1.2Y-3.1X+40Z \geq 0$

约束条件里不能出现变量之间的乘积，比如XY≥0，也不能出现根号之类花哨复杂的运算，尽管你可能想用它们。这条限制确实大大束缚了我们，但是从计算角度来看，线性规划模型正是靠这样的线性约束条件才能实现协同运作。

线性规划模型的第三大要素则是明确的目标，这为评估不同解的优劣提供了依据。Dantzig要求目标明确，从而迫使管理人员准确描述预期目的，他本人认为这一点是自己最大的实用成就之一。在指明目标时，我们假定活动的价值正比于其强度，因此要求目标函数是模型中各变量的线性表示，并在活动强度取尽所有可能值时求出目标函数的最大值或最小值。目标函数取得所求最值时，相应各变量的赋值就是线性模型的最优解。

线性规划的整体思路大致就是这样，但是这尚不足以解决Hitchcock和Wood提出的挑战。在建立问题模型之后如何给出其最优解？Dantzig还需要一种方法。这一次，他同样交上了答卷：单纯形算法（simplex algorithm）。这种计算方法吃进去的是数据，挤出来的是最优解。无论是一般应用还是专门解决TSP，单纯形算法都实在是太重要了，三言两语肯定讲不完，因此讨论就留到下一节再详细展开吧。

5.1.2 引入产品

前面介绍了许多概念和定义，因此现在举个简单的例子可能会有好处，这道小例题可以演示清楚这一切是如何组合作用的。产品是经济学家的心头最爱，我们也在模型中引入这一概念。为简便起见，假设只生产三种产品，分别称为产品*A*、产品*B*和产品*C*，它们的量也分别相应使用变量*A*、*B*、*C*表示。生产产品需要输入两种原料，这里假设它们是镍和钢，库存量分别为100磅（约45千克）和200磅（约91千克）。生产产品*A*需要消耗3磅镍和4磅钢，*B*需要3磅镍和2磅钢，*C*则需要1磅镍和8磅钢。

整个生产销售过程创造的总利润为$10A+5B+15C$，换言之，每单位产量的产品*A*盈利10美元，单位产量的*B*盈利5美元，单位产量的*C*则盈利15美元。试问，怎样安排三种产品的生产量，才能在不超出原料库存量的情况下赚到最多的钱？比如，假如我们集中生产最赚钱的产品，也就是*C*，那么产量到达25个单位时就把钢用完了，这样总利润就是375美元。不过线性规划可以告诉我们怎样才能赚得更多。这道题的线性规划模型如下所示，其中≤是小于等于号。

目标：$10A+5B+15C$取最大值

约束条件：

$3A+3B+1C\leq 100$ （镍）

$4A+2B+8C\leq 200$ （钢）

$A\geq 0, B?\geq 0, C\geq 0$

第一个约束条件保证我们的生产方案有充足的镍原料，第二个则保证有充足的钢原料。

这个线性规划模型的最优解是生产30个单位的产品*A*和10个单位的*C*，完全不生产*B*，这样总利润是450美元。这个结果相当简单，就算不用单纯形算法也能解出来。但是实际的生产问题复杂得多，变量和约束条件的数目可以多达几十万，这种情况下你绝对不会靠心算求出答案。

5.1.3 线性的世界

1948年，在威斯康星大学的一场会议上，Dantzig公开了通用线性规划模型及其求解所用的单纯形算法。这是他人生的决定性时刻。后来，他对这场报告津津乐道，经常说起当时的情形。翻来覆去地讲同一个经典故事，便可能导致不少细节在岁月里变味，这段往事也不例外。不过所有的讲述版本都抓住了故事的精髓：在一大群德高望重的数学家和经济学家面前，他是个前途无量的年轻后生，而他内心却忐忑不安。[1]在

报告后的讨论环节，Harold Hotelling站起身来，只说了一句话：“可我们都知道世界不是线性的。”无论从学术水平还是从身材上看，他都算是重量级人物。面对这么狠的批评，Dantzig一下子无言以对。[2]

1. Cottle, R. W. 2006. Math. Program. 105, 1–8.

2. Dantzig (1991).

> 突然听众中又有一人举起了手，那是冯·诺依曼。他说：“主席，主席，如果报告者不介意的话，我愿意替他回答。”我当然求之不得。他接着说：“报告者把题目定为‘线性规划’，陈述原理的时候也很谨慎。你的应用要是满足他的原理，那就用他的模型；要是不满足，那就不用呗。”

这尽管是个复杂的世界，但是很幸运，线性模型确实可以详尽描述多数复杂之处。据Dantzig在斯坦福大学的同事表示，他的办公室外面挂了一幅漫画，生动地总结了Dantzig、Hetelling和冯•诺依曼的这段往事。[3]漫画主角是《花生漫画》（*Peanuts*）人物Linus（莱纳斯），他攥着小毯子，吮吸着大拇指，摆出他的经典造型。配图文字写道：“幸福就是假设世界是线性的。”[4]

3. Gill, P. E., et al. 2008. Discrete Optim. 5, 151–158.

4. 这句话化用了漫画中的著名台词。原句为“Happiness Is A Warm Puppy”（幸福就是一只温暖的小狗），同时也是该系列第一本书的书名，小狗指的是《花生漫画》中的著名角色史努比。——译者注

5.1.4 应用

Linear Programming and Extensions（《线性规划及其推广》）是George Dantzig的经典著作，开篇第一句话写道：“检验一个理论的终极标准是它解决原始背景问题的能力。”[1]接下来是厚达600页的长篇大论，这真是开门见山。60年的实践已经反复证明，他的线性规划理论和相应的单纯形算法非常完满地通过了终极检验。

1. Dantzig (1963).

线性规划在工业界用途极广，令人惊叹，你说得上来的任何部门几乎都有它的用武之地。显然，通过线性规划安排生产，每天能节省数不清的自然资源，不过没有办法量化。线性规划也能省钱，Gina Kolata在《纽约时报》上撰文指出：“解决工业上的线性规划问题，是涉及每年上百亿美元的大事业。”[2]Hotelling教授，这话您听见没？

2. Kolata, G. 1989. *New York Times*, March 12.

至于如何用线性约束条件捕捉问题的实质，读者如果有兴趣了解，可以阅读Paul William的*Model Building in Mathematical Programming*（《数学规划中的建模》）[3]。这是一本好书，书里的例子有食品加工、精炼厂最优化、机票定价、种地、采矿、发电……五花八门，无所不包。

3. Williams, H. P. 1999. *Model Building in Mathematical Programming*. John Wiley & Sons, Chichester, UK.

5.2 单纯形算法

在《纽约时报》的封面上，要等多久才能看到一回数学的身影？费马大定理的证明成功登上了封面，而四色定理却不够格。至于研究线性规划的人，他们会自豪地说起两期封面故事。

> 一位名不见经传的苏联数学家做出了意外的发现，震惊了数学和计算机分析的世界。现在专家学者已经开始探索这一发现的实际应用。

——Malcolm W. Browne，《纽约时报》，1979年11月7日

> 方程系统常常变得数目繁杂，极为复杂，最强大的计算机也无法解决。如今，贝尔实验室有一名28岁的数学家在这方面实现了惊人的理论突破。

——James Gleick，《纽约时报》，1984年11月19日

上述两段评论来自两篇封面文章，分别介绍了求解线性规划问题的一种新算法，而且都是多项式时间算法。前一种算法由前苏联数学家Leonid Khachiyan发明，后一种则由贝尔实验室的Narendra Karmakar设计。

在这两篇文章中都咄咄透露出一点，Dantzig的单纯形算法在求解大规模问题时不甚实用，因此即将让出线性规划算法的王座。然而它并不会这么轻易就被推翻。20世纪80年代晚期，人们实现了新的单纯形算法程序，比如莱斯大学的Bob Bixby开发了CPLEX程序，IBM公司的John Forrest开发了OSL程序，这些都证明了单纯形算法仍有相当大的力量深藏不露。事实上，2000年评出的“世纪十大算法”（The Top Ten Algorithms of the Century）中，它的名字赫然在列，而且它至今仍在数学最优化领域不断立下汗马功劳。[1]

1. Dongara, J., F. Sullivan. 2000. Comp. Sci. Eng. 2, 22–23.

5.2.1 主元法求解

一代应用数学家与工程师领会了单纯性算法的实质，他们写了许多优秀的文献，用五花八门的方式描述算法的详细步骤。Vašek Chvátal的著作《线性规划》（*Linear Programming*）[1]渊博美妙，其他作品无出其

右者。所以，下面我们就借助产品制造的例子，一起理解他介绍的计算步骤。

1. Chvátal, V. 1983. *Linear Programming*. W. H. Freeman and Company, New York.

向线性规划模型中引入产品以后，变量*A*、*B*、*C*就表示商品*A*、*B*、*C*的生产活动强度。这些数值足以彻底明确问题，但如果要向管理部门提交报告，我们可能希望在报告里附加其他信息，包括总利润*Z*、镍的剩余库存量N和钢的剩余库存量*S*。这三个新变量定义如下：

$$Z = 0 + 10A + 5B + 15C$$

$$N = 100 - 3A - 3B - C$$

$$S = 200 - 4A - 2B - 8C$$

上述各式相当于用来查找这些变量数值的字典。

由这个字典可以自然想到一种赋值方式，即令*A*、*B*、*C*均为0，则由字典读出利润*Z*=0，镍存量*N*=100，钢存量*S*=200。这结果真令人失望，不过也很容易提升。以*A*为例，因为在利润的定义式里出现了10*A*这一项，而目前*A*取值为零，所以只要提高*A*的取值，那么公司就会有更多钱入账。不过在模型的约束条件下，*A*的产量最多可以提高到多少呢？保持*B*=0和*C*=0不变，就能满足最后三条取值非负的约束条件，不过我们还需要保证原材料镍和钢的库存够用，也就是*N*≥0且*S*≥0。*N*≥0意味着100/3*A*≥0，换言之，*A*最多可以取值$33\frac{1}{3}$，再多就没有镍可用了。同理，*S*≥0意味着*A*最多可以取值50，然后钢也就耗尽了。在这个例子中，镍就是最严格的约束因素，也就是紧约束，于是我们决定，令

$$A = 33\frac{1}{3}, B = 0, C = 0$$

总利润为$33\frac{1}{3}$美元。[2]

2. 在线性规划模型里，必须假定允许生产、销售非整数的产品量。本章结尾将继续讨论这点。

下面，让我们看看现在取值都是零的*B*和*C*。要如何增加它们的产量呢？这次结果并不是一目了然的，因为*A*已经用尽了所有的镍存量，所以在增加*B*和*C*的同时，必须减少*A*才能有镍可用。为了明确*A*对*B*、*C*的影响，可以把*A*移到字典定义式的左端，改写上述等式。同时，由于*N*目前取值为零，我们要把它也移到右端。

字典中对N的定义式为

$$N = 100 - 3A - 3B - C$$

把*A*移到左端，*N*移到右端，等式两端同时除以3，就得到等价的表示式

$$A = 33\frac{1}{3} - \frac{1}{3}N - B - \frac{1}{3}C$$

我们会在新的字典中使用*A*的上述新定义式，并且将它代入总利润*Z*和镍存量*S*的定义式中，从而得到方程组

$$Z = 333\frac{1}{3} - 3\frac{1}{3}N - 5B + 11\frac{2}{3}C$$

$$A = 33\frac{1}{3} - \frac{1}{3}N - B - \frac{1}{3}C$$

$$S = 66\frac{2}{3} + 1\frac{1}{3}N + 2B - 6\frac{2}{3}C$$

这步操作互换了*A*和*N*的地位，称为选主元（pivoting），其中*A*称为换入变量（incoming variable），*N*则称为换出变量（leaving variable）。请注意，虽然变量*Z*和*S*的定义式不同于之前字典中的形式，但是*Z*和*S*的意义依然不变，分别代表总利润和镍存量。事实上，在改写变量定义式时，由于使用的等式对任何赋值都成立，因此变量的意义都不会改变。

得到新字典之后，很自然的答案是令*N*、*B*、*C*均为零，读出其他各变量的值为

$$Z = 333\frac{1}{3}, A = 33\frac{1}{3}, S = 66\frac{2}{3}$$

$33\frac{1}{3}$美元的总利润肯定比零利润多，但是我们完全可以赚更多。仔细观察改写后的利润定义式，可以发现，N和B这两项的系数都是负的，因此，如果增加其中任何一个的量，就会导致总利润下降。幸好C项的系数是正的，所以下一步换主元操作时，我们就应该考虑增加C的值。

下面，让我们把*C*作为换入变量，逐步进行换主元操作。首先，保持*N*和*B*为零，根据新字典，有

$$A = 33\frac{1}{3} - \frac{1}{3}C \geqslant 0$$

由此可知，C最多可以增加到100。类似可得

$$S = 66\frac{2}{3} - 6\frac{2}{3}C \geqslant 0$$

由此可知，C最多可以增加到10。又因为10<100，所以S是这一步换主元操作的换出变量。

下一步是改写字典，用S表示C并依次代入各式，具体算术过程在此略去。最终，新的方程组为

$$Z = 450 - N - \frac{3}{2}B - \frac{7}{4}S$$

$$A = 30 - \frac{2}{5}N - \frac{11}{10}B - \frac{1}{20}S$$

$$C = 10 - \frac{1}{5}N - \frac{3}{10}B - \frac{3}{20}S$$

对于刚得到的新字典，自然的答案是令

$$N = B = S = 0, Z = 450, A = 30, C = 10$$

这种赋值方式就是我们在前一节宣称的最优解。你不必盲目相信我的断言，因为我们可以给出充分的证据：利润式Z=450−N−32B−74S证明了，生产商绝对不可能获得高于450美元的利润。事实上，无论如何赋值，N、B和S都必然是非负数，因此总利润就是450美元减去一个非负数。以上，证明完毕。

可以看出，单纯形算法就是这样，通过换主元操作把一部字典变成另一部字典，每一步变换都希望增加目标函数的值。如果到了某一部字典，目标函数定义式可以证明，自然解就是最优解，那么算法就终止。易如反掌，不过刚才我们有意无视了几处难点。一般来说，如何找到初始字典？如何确定算法必定终止？这些问题都可以获得解答，Chvátal对此做了详尽的分析，感兴趣的读者请参阅他的书。

如果你有意求解其他的题目，但又讨厌人工完成这么多运算，有一个网页提供了Bob Vanderbei的单纯形换主元工具（Simplex Pivot Tool）[3]，请上网一探究竟。这个小工具允许你以字典的形式建立起自

己的线性规划模型，让你选择换入和换出变量，实现换主元操作。假如你犯了错误，选择的换出变量与紧约束条件不对应，该网页甚至会提醒你这一点。

3. http://campuscgi.princeton.edu/～rvdb/JAVA/pivot/simple.html。这是下书的相关工具：Vanderbei, R. J. 2001. *Linear Programming: Foundations and Extensions*. Kluwer, Boston, MA.

5.2.2 多项式时间的选主元规则

Dantzig给出了一个算法，但我们凭什么相信它是可行的，无论多大规模的题目都能解决？问得好。

2009年，计算机科学基础年会（IEEE FOCS）迎来50周年，并在亚特兰大举行了庆祝会议。Richard Karp以“优秀的算法好在何处？”为题，做了开场报告。他举了一堆优秀算法的例子，单纯形算法排在第一个，上榜理由是实际高效可行，加之应用广泛。不过Karp当时也不得不指出，在复杂度方面，单纯形算法并不优秀。事实上，它并不能保证在多项式时间内完成运行。

问题在于，虽然可能的字典数是有限的，但是这个有限的数字却是指数级的。目前人们并不知道，能否按照某种方式选择每步的换入变量和换出变量，使得只需经过多项式次数的换主元操作，就能得到最优字典。实际上，人们已经知道，有好几种思路自然的换主元规则并不是多项式的，也就是说已经针对它们构造出了特定的极端反例，导致单纯元算法按照相应的换主元规则运行时，换主元的步骤达到了惊人的指数级。

既然如此，Dantzig本人对单纯形算法的巨大信心从何而来？答案是，他其实并没有信心，至少一开始没有。[1]

1. Albers and Reid (1986).

> 刚开始，我觉得这个方法有可能效率不错，但不一定实际可行。如果题目很大，就可能有很多种组合（顶点，corner point，也称为角点或极点），说不定跟天上的星星一样多，那算法可能就需要上百万步才能解出来。这个步骤数确实比可能组合的总数要小，所以可以认为算法是高效的，但基本还算不上可行。因此，我当时就继续去寻找更好的其他算法了。

直到国防部的同事们成功解决了一道又一道难题之后，Dantzig才终于认可了自己的算法。

时至今日，情况依然如此。单纯形算法是如今数学中应用最广的工具之一，但是我们并不清楚，今后从实际模型中涌现出的问题是否一直能够迎刃而解。如果有人能回答这个问题，绝对能成为《纽约时报》的封面故事主角。如果有人能提出新的选主元规则，保证在多项式次数的步骤之内获得最优解，那么声望乃至财富都将从天而降，滚滚而来。

5.2.3 百万倍大提速

Dantzig在计量经济学会1948年年会上发表了报告，摘要只有寥寥几行，最后一句总结道："对于由约翰·冯·诺依曼或笔者等人提出的此类计算方法，在实现规划问题的解法时，建议配合使用大规模数字计算机。"[1]当时，数字计算机时代刚刚拉开序幕，应用数学家已为此激动万分，Dantzig亦把单纯形算法推到了最前线，等待可能的计算机实现。

1. Dantzig, G. B. 1949. Econometrica 17, 74–75.

然而，单纯形算法的第一次大规模测试却没有用到计算机。1947年，这场计算测试在美国国家标准局举行，一群测试人员手动操作台式计算器，总计连续工作120个工日。[2]测试实例的原始问题是选择成本最低的日常健康饮食计划，模型包括9个约束条件和77个非负变量。在如今的工业线性规划模型中，约束条件和变量都数以十万计。当年的测试实例虽然规模挺大，相比之下也只是小巫见大巫而已。幸运的是，从当年的计算饮食计划问题到发布新型线性规划软件之间的这些年里，人们殚精竭虑，致力于调整改进单纯形算法，为程序实现做准备。

2. Dantzig, G. B. 1982. Oper. Res. Let. 1, 43–48. 工日是工作时间的计算单位，一个人工作一天记为一个工日。——译者注

在单纯形算法的发展史上，有一个阶段格外重要。1984年，Karmarkar公布了内点法（interior-point method），这是一种具有竞争力的线性规划新算法。围绕内点法的新闻报道使线性规划成为了舆论关注的焦点，性能强劲的台式工作站和个人电脑恰恰也在此时走入寻常百姓家。强强联手，线性规划世界自此跨进超光速发展时代。事实上，在1987到2002年间，基于单纯形算法的求解程序速度加快了百万倍。Bob Bixby具体解释道："机器运行速度提高了三个量级，算法运行速度又提高了三个量级，合起来就让解题能力提高了六个量级。10年前需要花一整年才能解决的题目，如今不到30秒就能解决。"[3]这下，线性规划解题性能获得了可观的提升。后来的岁月风平浪静，波澜不惊，但是我们完全有理由相信和期待，研究活动继续开展下去，必将掀起新一轮突飞猛

进的浪潮。许多科研项目得到了广泛的资助，尤其是来自美国国家科学基金会和海军研究办公室的大力支持。这些课题致力于促成单纯形求解程序的更新换代，新一代程序将利用未来计算机芯片的众核性能。如果历史可以为鉴，那么线性规划研究者将面对更好的硬件，并在此基础上设计出更好的实现，让Dantzig的单纯形算法不但能更快地作出解答，而且能解决更大规模的题目。

3. Bixby, R. E. 2002. Operations Research 50, 3–15.

5.2.4 名字背后的故事

单纯形算法的英文是simplex algorithm，感觉像是用simple（简单、单纯）这个词新造出来的生词一样，很有近年流行的文字游戏的味道。实际上，用谷歌搜索这个英文单词，返回的前几条结果确实如此。但是它其实是个几何学概念，表示一种经典的几何形体：单纯形，即有n+1个顶点的n维多面体（可剖分空间）。二维单纯形就是三角形，而三维单纯形则是四面体。当时，Dantzig的好友Theodore Motzkin建议，借用这个几何概念为算法命名。"'单纯形方法'这种说法来自我和T. Motzkin的一场讨论。他觉得如果考虑列向量的几何意义，那么可以恰当地把我的方法描述为相邻的单纯形之间的变换。"[1]他采纳了这个建议，这实在是太遗憾了，因为"Dantzig算法"明明是个不错的名字，可以恰如其分地纪念他的杰出贡献。

1. Dantzig (1991).

5.3 买一赠一：线性规划的对偶性

最优字典能够给出证据，证明单纯形算法已经给出了最优解，但是要说服经理相信这一点，恐怕没那么轻巧。实际上，大概没人打算把整个换主元操作序列都交给经理过目，让他一步一步验算；同样，也没法要求检验Bob Bixby的线性规划解题工具，看看它那上百万行的代码有没有问题，以确认它正确地实现了Dantzig的算法。我们需要的其实是得出字典证据的捷径，而在线性规划中，对偶性（duality）的作用正在于此。

让我们回到带有三种产品线性规划的模型里，观察最终的利润式

$$Z = 450 - N - \frac{3}{2}B - \frac{7}{4}S$$

这个表达式是由Z的初始定义与所有允许赋值都满足的若干等式相加而得的。我们知道这句话是事实，可惜经理并不知道。没关系，别担心。如果我们在总利润Z的最终定义式里，把N和S在初始字典中的定义式分别代入，那么就重新得到了总利润Z的初始表达式。这样一来，经理应该可以相信，我们没有做什么手脚。不过我们可以更简洁地完成这一过程，直接证明盈利不可能超过450美元。

为了建立证明，我们首先取镍存量N和钢存量S的初始约束条件，再分别乘以总利润Z的最终定义式中相应项的系数（不带负号），得到

$$1 \times (3A + 3B + 1C \leqslant 100)$$

及

$$\frac{7}{4} \times (4A + 2B + 8C \leqslant 200)$$

然后计算得两个新的不等式，相加得到一个不等式

$$10A + 6\frac{1}{2}B + 15C \leqslant 450$$

经理看到这个，肯定也会点头，同意任何可能的产品生产水平都满足上式。而与总利润$10A+5B+15C$相比，可以看出，盈利不可能超过450美元，就算把产品B的利润率（profit margin）提高到每单位6.5美元也无济于事。稍做几步乘法和加法，即可得出结论：我们提出的生产方案是最优的。

下面，我们回顾一下刚才证明中的几处要点。首先，我们取了N项系数和S项系数，称为y_N和y_S，它们都是非负数，因此可以用来与镍和钢的约束条件相乘。其次，相乘之后得到了两个新的约束条件，两式相加时，A、B、C前面的系数都不小于相应产品的单位利润值。

这两条规则确保我们对y_N和y_S的使用有理可依，而它们事实上正是另一道线性规划问题的约束条件。

目标：$100y_N+200y_S$取最大值

约束条件：

$$3y_N + 4y_S \geqslant 10$$

$$3y_N + 2y_S \geqslant 5$$

$$1y_N + 8y_S \geqslant 15$$

$$y_N \geqslant 0,\ y_S \geqslant 0$$

上面三个约束条件分别对应于变量A、B、C，它们可以保证，在原始不等式乘以y_N和y_S之后，得到的新不等式中，各项产品前的系数都至少不低于三种产品的单位利润值。如果y_N和y_S的赋值满足以上各条约束条件，就可以说明总利润值不会超过$100\ y_N + 200\ y_S$。因此，为了获得有说服力的证据以供给经理看，我们希望想办法使这个量取得最小值。

这个新的线性模型称为对偶问题（dual problem），按照Dantzig的父亲Tobias的建议，原始问题则称为原问题（primal problem）。对偶线性规划问题中，约束条件要求，任何满足条件的对偶变量赋值都给出原目标函数的一个上界或下界。本例中，有 $10A + 5B + 15C \leq 100y_N + 200y_S$。

单纯形算法的最优字典给出，取值

$$A = 30,\ B = 0,\ C = 10,\ y_N = 1,\ y_S = \frac{7}{4}$$

使上面的不等式两端均为450，这就证明了450是原目标函数的最优值，也是对偶目标函数的最优值。一举两得，花一道题的工夫，解出两道题的答案，多划算啊。

值得注意的是，对于任何线性规划题目，都存在这样简洁优美的最优性证明：单纯形算法构造出了用来相乘的系数，它们把线性规划原问

题的约束条件合而为一，从而强有力地说明目标函数不可能取更大的值。进一步，这些系数本身又是线性规划对偶问题的最优解，而且原目标函数和对偶目标函数两者取值相等。这一美妙的结果称为强对偶性定理（strong duality theorem），最初由约翰·冯·诺依曼叙述并证明。[2]

2. 准确叙述为，如果一道线性规划问题与其对偶问题都有可行解，则它们都有最优解，且它们的目标函数最优值相等。

强对偶性是线性规划问题里的重要理论，不过在TSP论题里，我们确实只需要更简单的命题，也就是弱对偶性定理（weak duality theorem）：在线性规划中，对偶问题的任意解都给出原目标函数的一个界。刚才几页草草掠过，许多细节浅尝辄止，你若错过了几处，也不必忧虑，下一节我们将针对TSP这种线性规划的特殊情形，直白易懂地解释，究竟何为对偶性，如何用线性不等式布下天罗地网，把旅行商团团围住。

5.4 TSP对应的度约束线性规划的松弛

线性规划方法问世之后，没过多久就打入了旅行商的世界。Dantzig与Hotelling和约翰·冯·诺依曼的那段往事发生在1948年9月9日，而在次年秋天，Julia Robinson就公布了第一个基于线性规划的TSP解法。

在表面上看来，TSP似乎与Dantzig发明的通用经济统筹方法并不契合。事实上，如果你把五名TSP研究者关在一间房子里，让他们解释为什么线性规划方法是TSP的自然求解思路，最后很可能会得到五种截然不同的答案。我个人最欣赏的线性规划的理解方式如本章开篇一段所述，也就是将所有路线都满足的若干简单基本规则结合在一起，从而获得路线的品质保证。

在对偶性的脚本上，这一切写得明明白白：藉由对偶变量的赋值，规则之间相互结合；而通过弱对偶性定理，品质保证便水到渠成了。

让我们看看实际应用如何吧。按照之前的惯常做法，我们考虑对称的TSP，也就是旅行成本与旅行方向无关的情形。读者想必已经很熟悉用来描述这种题目的图论术语了，顶点代表的是城市，边代表的是城市间的道路。路线就是选择的一组边，这些边合起来构成一条哈密顿回路，如图5-2所示，图中给出了24座城市的完全图，红线表示一条选定的路线。

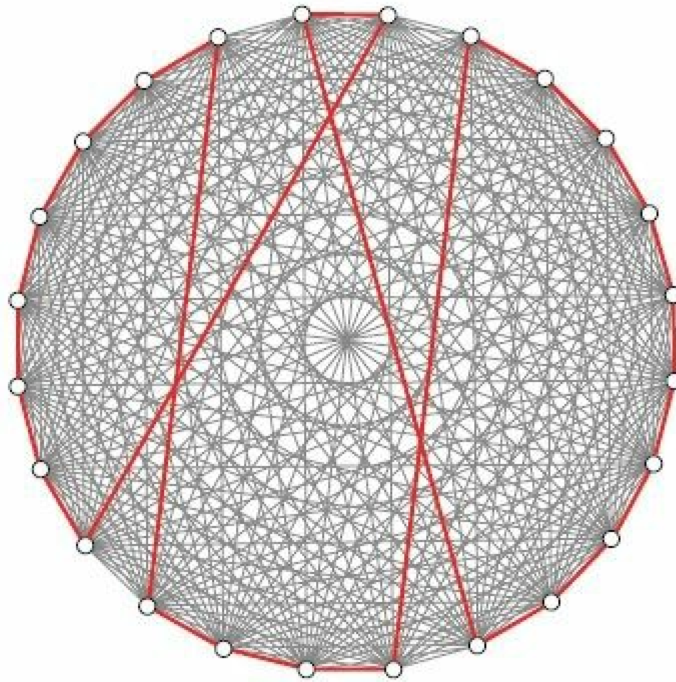


图**5-2**　完全图某条路线的边

画图的时候，完全可以用灰线和红线表示，但是用数学语言描述题目时，最好还是给边赋值0和1来表示两种情况：赋值为0的边不在路线里，而赋值为1的边在所给的路线中。

图5-2的例子有24座城市，因此有276个这样的值。数值太多了！如果要仔细考虑，我们难免会困在一大堆记号里动弹不得，所以我们先简化题目，看看更原始的图5-3。这里只有区区6座城市，为了确定一条路线，需要15个值。我们把连接每对顶点i和j的边所取的值记为x_{ij}，具体说来，这15个值分别为$x_{12}, x_{13}, x_{14}, x_{15}, x_{16}, x_{23}, x_{24}, x_{25}, x_{26}, x_{34}, x_{35}, x_{36}, x_{45}, x_{46}, x_{56}$。这些就是线性规划模型中的变量，它们量度的是路线是否用到了对应的边，如果你愿意的话，也可以把它们理解为某种“经济活动强度”。

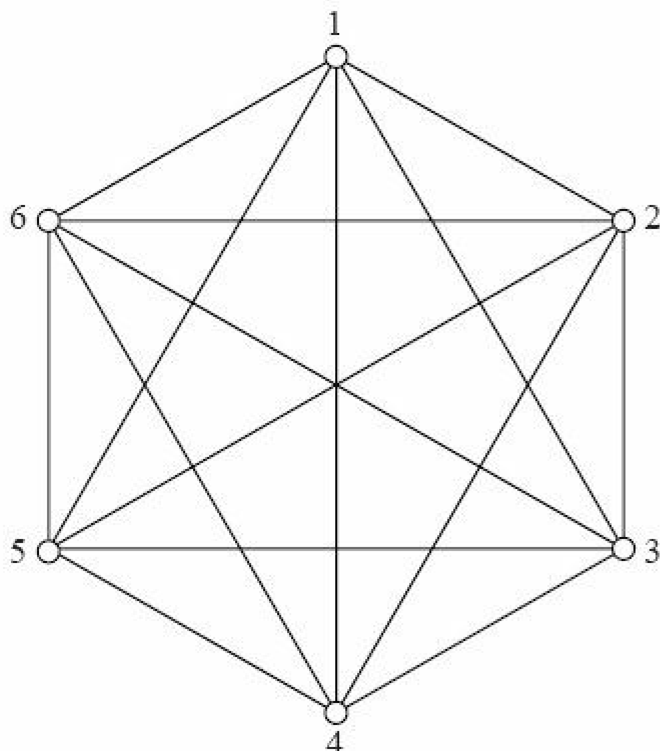


图**5-3　6**个顶点的完全图

我们把每对城市之间的旅行成本记为c_{ij}，这样一来，如果某条边在路线中，那么相应的c_{ij}乘以x_{ij}就等于c_{ij}，反之则该式取值为0，一条路线的总成本便可记为线性表达式$c_{12}x_{12} + c_{13}x_{13} + \ldots + c_{56}x_{56}$。这就是我们在TSP的线性规划模型中需要最小化的目标函数。至此，我们已经确定了变量和目标函数，那么约束条件如何得到呢？

哎呀，大事不妙，我们可没法直接说条件就是挑出一条路线作为解。要想使用线性规划方法，就必须严格遵照Dantzig建立的模型，严格使用线性规则。寻找合适的线性规则是我们的解法的核心之道。

5.4.1 度约束条件

开始求解时，疯帽匠指出，线性规划中的每个变量都必须是非负数。没错，可是这没什么实际效果。要想真正有所进展，我们必须仔细观察，图的边组成的子集中，能够形成路线的子集究竟有何特别之处，因为有的子集能够形成路线，有的却不能。如何利用线性约束条件区分这两种子集呢？

Julia Robinson的研究便是以此作为出发点的。她注意到，在任何路线中，图的每个顶点都恰好与两条边相连。这条规则本身并不是线性规则，不过由此可以推出，如果把经过给定顶点的所有边的x_{ij}值相加，那么得到的和必然恰好为2。因此，我们对任何一座城市都有一个度约束条件（degree constraint），从而得到线性规划模型。

目标：$c_{12}x_{12} + c_{13}x_{13} + \ldots + c_{56}x_{56}$取最小值

约束条件：

$$
\begin{aligned}
x_{12} + x_{13} + x_{14} + x_{15} + x_{16} &= 2 \\
x_{12} + x_{23} + x_{24} + x_{25} + x_{26} &= 2 \\
x_{13} + x_{23} + x_{34} + x_{35} + x_{36} &= 2 \\
x_{14} + x_{24} + x_{34} + x_{45} + x_{46} &= 2 \\
x_{15} + x_{25} + x_{35} + x_{45} + x_{56} &= 2 \\
x_{16} + x_{26} + x_{36} + x_{46} + x_{56} &= 2
\end{aligned}
$$

(i,j)是任意一对顶点，且$x_{ij} \geq 0$

上述模型称为TSP对应的度约束线性规划的松弛（degree LP relaxation）。

上述松弛的最优解本身往往未必是完整路线，即便如此，它依然能够提供宝贵的信息。事实上，每条路线都是线性规划的可行解，所以最优的线性规划目标函数值必然不会超过最优路线的成本，这是运用线性规划解决TSP时最需理解和领悟的重点所在。对于TSP对应的线性规划问题，我们在一大堆可行解中选择最优解，从而获得路线可能成本的一个下界，数值为*X*。任意路线的成本都绝对不可能低于*X*，就像之前我们确信产品利润绝对不可能超过450美元一样。

5.4.2 控制区

上述给出界限的思想非常重要，值得再次细细品味。所以，下面我们换种角度，重新审视度约束线性规划的松弛。这一次，我们直接研究对偶问题，采取由Michael Jünger和William Pulleyblank提出的思想，用一种漂亮的技巧处理TSP的几何题目。[1]在本节的说明中，我们假定TSP

题目的旅行成本是各点之间的直线距离。

1. Jünger, M., W. R. Pulleyblank. 1993. In: S. D. Chatterji et al., eds. *Jahrbuch Überblicke Mathematik*. Vieweg, Brunschweig/Wiesbaden, Germany. 1–24.

首先，假设以一座城市为圆心、以r为半径画一个圆盘，使圆与其他所有城市都不接触，如图5-4所示。无论路线如何，旅行商必定会在旅程中的某一刻到达这座城市，为此，他到达和离开这座城市时都必须至少经过距离r。我们由此可以得出结论，每条路线的长度至少为$2r$。进一步，我们可以以每座城市i为圆心、分别画一个半径为r_i的圆盘，只要它们互不交叉重叠即可，如图5-5所示。这样一来，任何TSP路线的长度都不短于所有圆盘半径之和的两倍，我们便得到了路线长度的一个下界。Jünger和Pulleyblank将这些圆盘命名为控制区（control zone）。

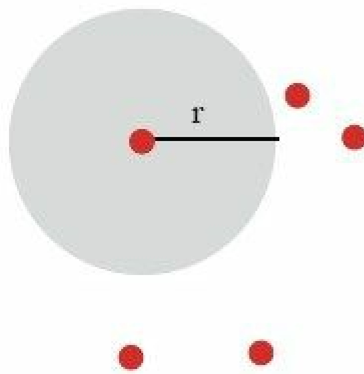


图**5-4**　**1**个控制区

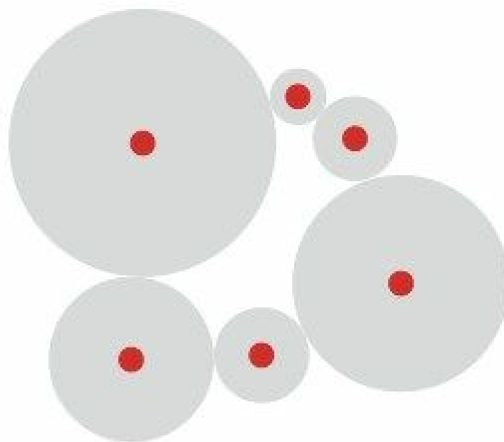


图**5-5**　**6**个控制区

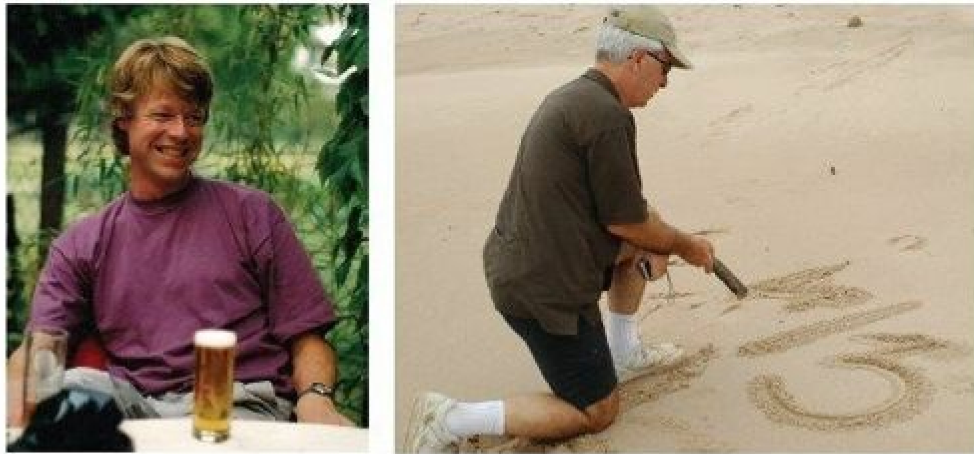


图**5-6　Michael Jünger**（左），**Regine Strobl**摄；**William Pulleyblank**（右），**Nick Harvey**摄

我们希望控制区给出尽量大的下界，为此，有必要适当选择各个圆盘的半径，在互不交叠的前提条件下，使圆盘半径之和的二倍取最大值。无交叉无重叠的前提条件可以简洁地表示为：对于每对城市i和j，它们对应圆盘的半径r_i与r_j之和不得超过城市之间的距离，即r_i与r_j必须满足

$$r_i+r_j \leqslant c_{ij}.$$

由此可见，为了获得最好的一系列控制区分布，要解决如下的线性规划题目。

目标：$2r_1 + 2r_2 + 2r_3 + 2\, r_4 + 2r_5 + 2r_6$取最大值；

约束条件：

$$r_1 + r_2 \leqslant c_{12}, r_1 + r_3 \leqslant c_{13}, r_1 + r_4 \leqslant c_{14}$$

$$r_1 + r_5 \leqslant c_{15}, r_1 + r_6 \leqslant c_{16}, r_2 + r_3 \leqslant c_{23}$$

$$r_2 + r_4 \leqslant c_{24}, r_2 + r_5 \leqslant c_{25}, r_2 + r_6 \leqslant c_{26}$$

$$r_3 + r_4 \leqslant c_{34}, r_3 + r_5 \leqslant c_{35}, r_3 + r_6 \leqslant c_{36}$$

$$r_4 + r_5 \leqslant c_{45}, r_4 + r_6 \leqslant c_{46}, r_5 + r_6 \leqslant c_{56}$$

满足上述模型的任何可行解都给出一组互不交叠的控制区分布，所以同时也给出原题的所有TSP路线长度的一个下界；最优解给出的控制区下界是最强的。

读到这个命题，你也许回想起了之前提到过的弱对偶性定理。的确，控制区分布问题正是度约束线性规划的松弛问题的对偶问题！何以见得？只需注意到，对偶问题中每个顶点i都有一个系数y_i对应于第i个度约束条件。当我们分别用y_i乘以相应的约束条件，将它们加在一起时，得到的线性表达式中，每个变量x_{ij}前的系数必然不会大于c_{ij}的值。由于变量c_{ij}出现在第i个和第j个约束条件中，因此，我们便要求$y_i+y_j \le c_{ij}$。这个式子正是上面的互不交叠条件，唯一区别只是半径r_i改用y_i表示而已。

当然，我们必须承认，这里有一点投机取巧了。实际上，对偶线性规划问题允许控制区的半径取负数值，原因在于度约束条件是等式而非不等式，所以用负数与等式相乘也是完全合理的。不过，我们只是为了严谨起见而指出这一点，它对结果并没有影响，因为虽然在上面的题目中并未直接限定取值为非负数，但是根据三角不等式可以证明，必然存在一组最优的对偶系数，使得每个系数取值都是非负数。

5.5 消去子回路

在TSP的线性规划解法中，下一步是引入一系列简单而强有力的附加规则。为此，我们再次考虑图5-5中包含6座城市的控制区分布示意图。该图中，控制区连成环状分布，很容易勾勒出一条路线，穿过各个控制区，完全不浪费任何空间。然而，一般情况下我们并没有这么幸运。如图5-7所示的现象更为常见，图中各个控制区挤成好几堆，因此勾连路线时不得不留出一大截空隙。新规则利用的正是这样的空隙。

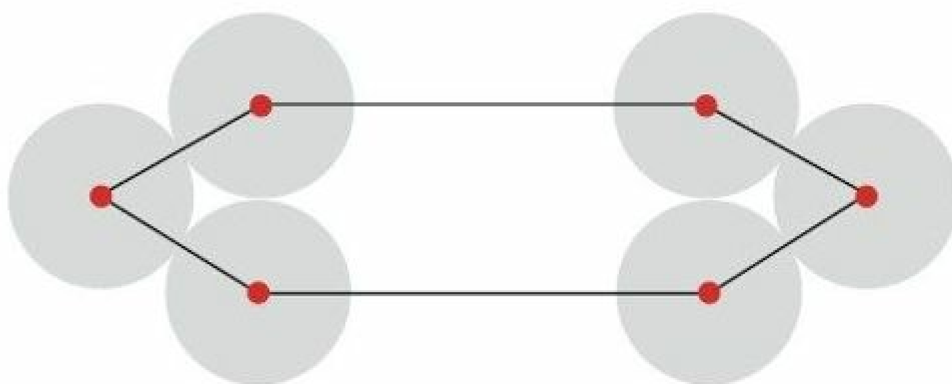


图**5-7**　不合适的控制区分布

下面我们描述一下新规则的几何思想。在控制区分布剩下的空隙中，可以围绕一堆点的团簇，画一条环带，如图5-8所示。任何一条路线必然在某一刻到达这一簇点，旅行商进出这组城市时，每一程都必须经过这个环带，所以总共至少要经过两次。因此，控制区给出的路线长度下界中可以加上环带最小宽度的两倍。Jünger和Pulleyblank把这样的环带称为“护城河”（moat），因为它们确实就像护城河一样。

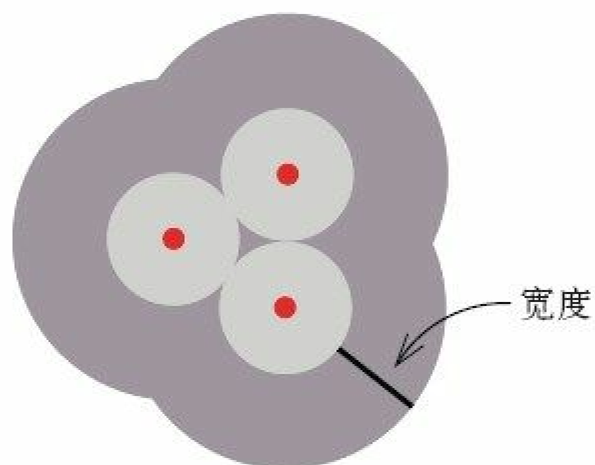


图**5-8**　护城河区域

如图5-9所示，在6座城市的题目中，可以使用两个护城河区域弥合控制区之间的空隙，这样一来，沿着城市勾画路线时，便同样不会浪费任何空间距离了。虽然我们并非次次都能如此幸运，充分利用一切边角空间，但是只要仔细安排护城河和控制区的分布，通常依然能够给出很强的路线长度下界。以图5-10为例，这里有100座城市，图中的分布很合理，足以证明给出的路线长度最多只比最优路线长0.65%。

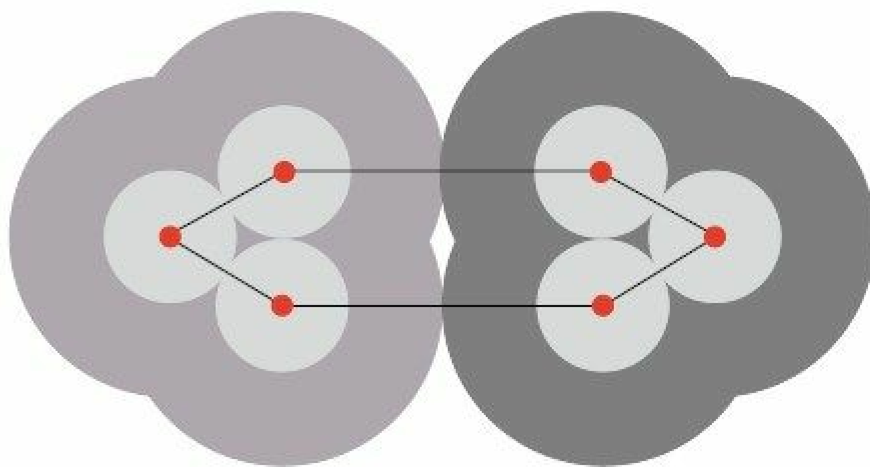


图**5-9**　用护城河弥合控制区之间的空隙

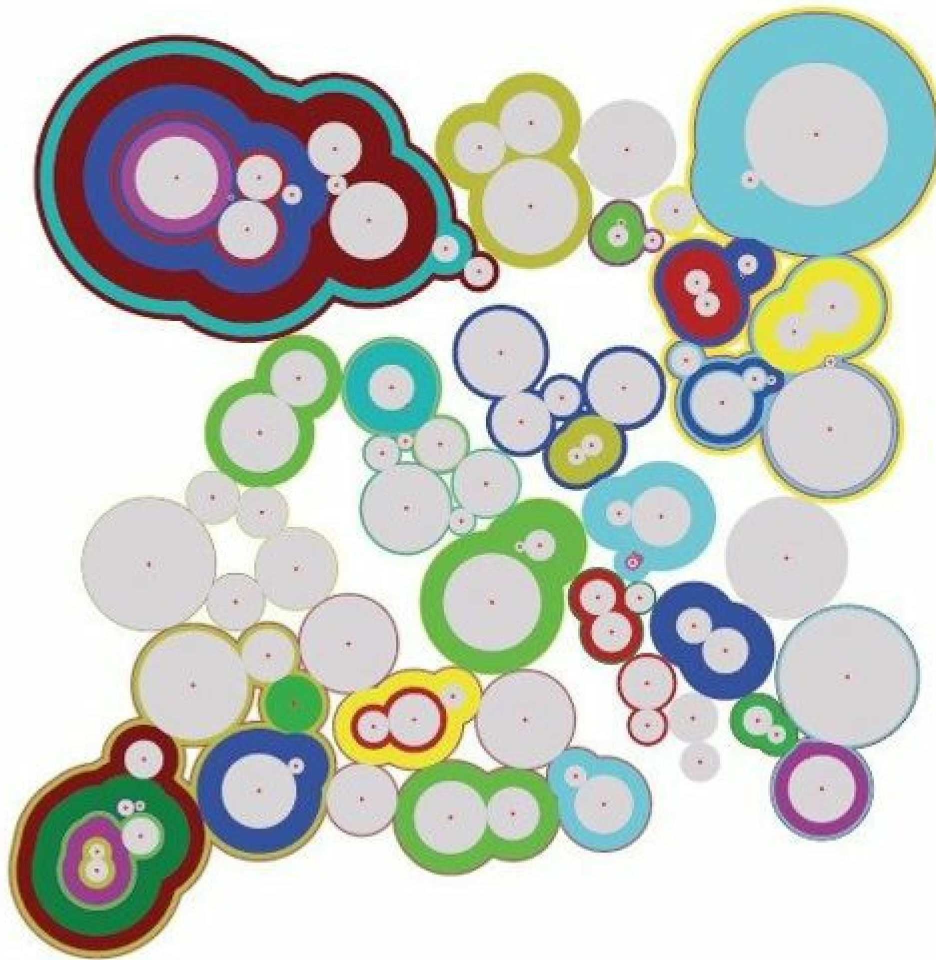


图**5-10**　护城河和控制区的综合分布

这个100座城市的例子很漂亮，不过如果想要更好地理解这种路线的下界，最好选择规模更小的题目实际演练。在此，我向读者推荐Geodual软件包，它是由德国科隆大学的Mike Jünger研究团队开发的。[1] 该软件能够为20座城市左右规模的TSP题目给出最优解，同时还会提供一幅优美的护城河和控制区分布图，成果一例见图5-11。

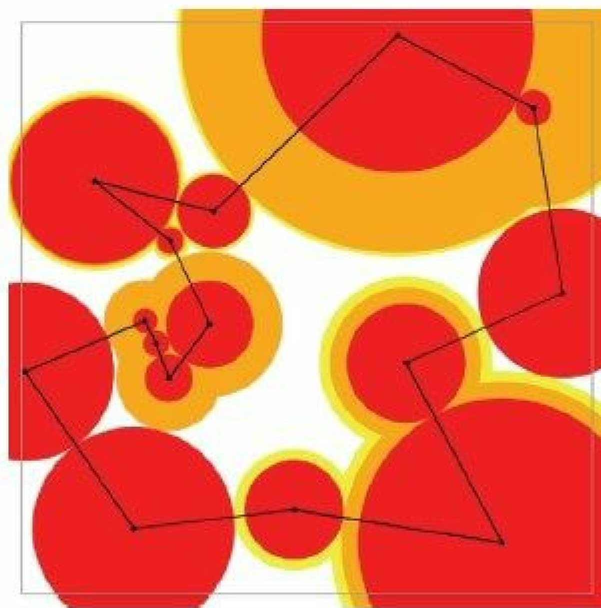


图**5-11　Geodual**求解**15**座城市的**TSP**时的真实截图

1. 参见http://www.informatik.uni-koeln.de/ls_juenger/research/geodual/。

5.5.1 子回路不等式

在一些TSP题目中，我们如果想用原始解构造一条路线，就会遭到度约束线性规划的松弛的无情捉弄。图5-7的6座城市TSP便是如此。事实上，考虑到跨过巨大空隙的边会有高昂的旅行成本，单纯形算法给出的解会包含两个三角形，而不是一次性走过全部点的完整回路。显然，这虽然是松弛问题的可行解，对旅行商而言却并不可行。

注意到有些边跨越城市团簇之间的空隙，如图5-12的绿色线段所示，而所有路线都必定包含至少两条这样的边，于是我们能直接做出简便的补救。仿照Jünger和Pulleyblank构建护城河的思路，我们可以提出一条规则：对应这些跨团簇的边的变量之和至少为2。

$$x_{13}+x_{14}+x_{15}+x_{23}+x_{24}+x_{25}+x_{63}+x_{64}+x_{65}\geqslant 2$$

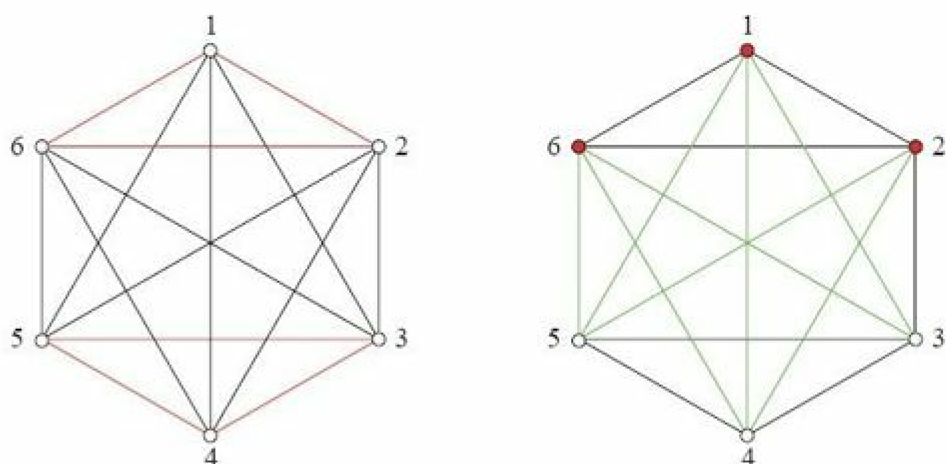


图**5-12**　构成子回路不等式的边

有了这个不等式，再加上度约束条件，图5-12的解便不可能包含左图中红边组成的子回路了。因此，这条规则称为子回路消去约束条件（subtour-elimination constraint），简称子回路不等式（subtour inequality）。

对于全部城市的任意适当子集*S*，因为一端在*S*中而另一端不在*S*中的边所对应的变量之和至少为2，可得到一个子回路不等式。图5-12中有*S*={1, 2, 6}，而图5-13展示的例子规模更大，落在矩形框内的顶点构成了集合*S*，而染成绿色的各边对应于相应的子回路不等式中的各个变量。

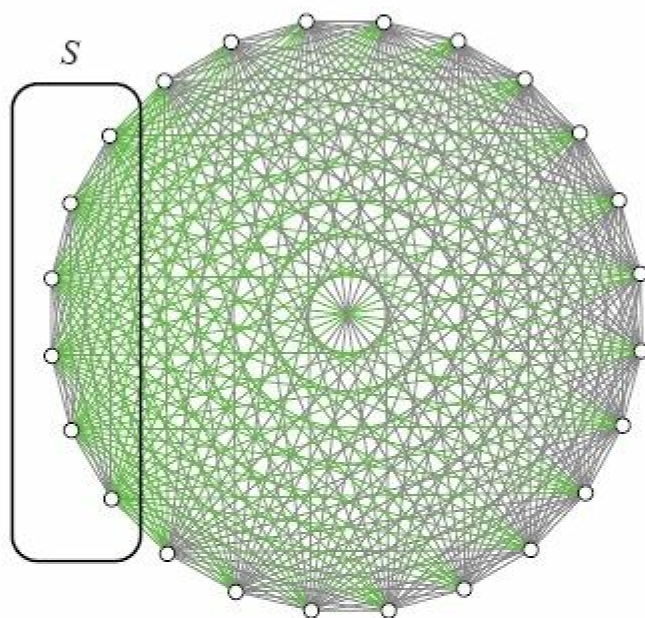


图**5-13**　构成子回路不等式的绿边

这些附加规则虽然简单，但通过线性规划结合在一起时却能带来巨大的威力。度约束线性规划的松弛和所有的子回路不等式联合起来，构成的模型称为子回路线性规划的松弛（subtour LP relaxation）。事实上，这种方法可以获得优质的路线长度下界，其结果对实际应用线性规划成功求解TSP举足轻重，功不可没。对于随机生成的几何题目，子回路方法给出的下界几乎每次都与最优路线相差不到1%。如果选取42座城市的环游美国数据集为例，可以看到，得到的路线长度下界是697个单位长度，而最优路线为699个单位长度。在这一特例中，与理论的品质保证相比，实际的最优路线仅仅超出0.3%而已。

5.5.2 “4/3猜想”

总体上，我们并没有这么强的品质保证，幸好，糟糕的反例看来只是偶然例外，优良的范例才是正常规则。对于满足三角不等式的题目来说，如何才能判断路线长度下界质量是好是坏？这是一个颇有意味的问题。从乐观的一方面来看，人们已经知道，最优路线的成本必然不会超过子路线下界的3/2倍。比起前一章介绍的Christofides定理，这个理论结果很出色。

但是凡事都有两方面，这个问题的负面则在于，人们已经发现了一系列题目，当城市数目增大时，它们的最优路线成本与子路线下界之比将逐步逼近4/3。4/3这个数字就是最差的可能性吗？这个问题称为“4/3猜想”（4/3rds Conjecture），加拿大渥太华大学的Sylvia Boyd是钻研该问题的顶尖专家。她与同事通力合作，已经验证确定，在不超过十座城市的所有情形下，该猜想都成立。她还提出了一个更尖锐的问题，有可能为完整解决猜想、证实最终结论提供必要的思考方向。[1]

1. Benoit, G., S. Boyd. 2008. Math. Oper. Res. 33, 921–931.

品质保证从3/2升级到4/3，看似只是一小步改进，但基本上，要想迈出这一小步，必须深刻理解子回路线性规划的松弛问题的可行解结构。[2]反过来，这样的理解也会产生新的TSP方法，提出新的解题规则，从而有机会推动实际计算超越现有的极限。

2. 麻省理工学院的Michel Goemans已经证明了若干与子回路LP松弛之解有关的重要结论。

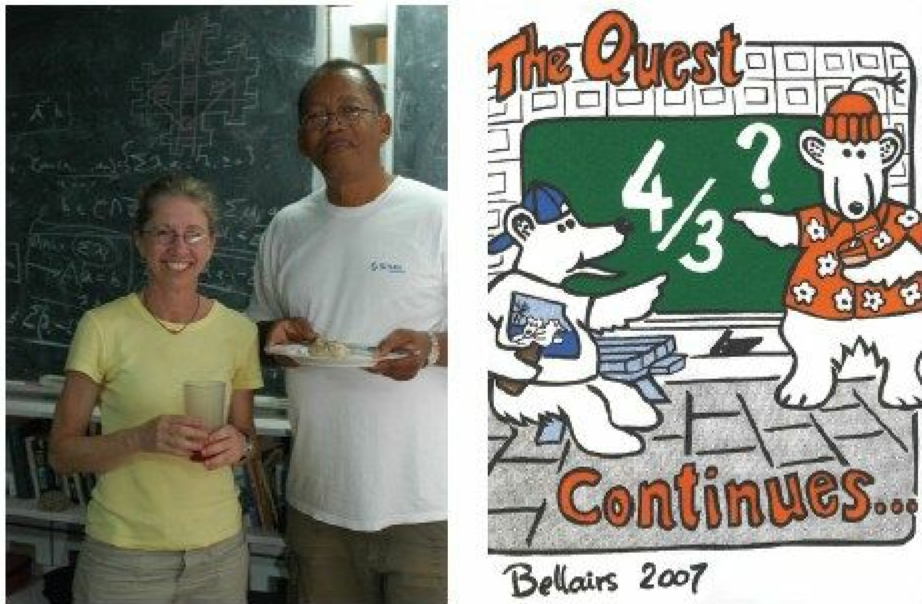


图**5-14　Sylvia Royd**和**Richard Haynes**（左），**Bellairs Workshop 2007**，**Nick Harvey**摄；**“4/3**猜想**”**文化衫图案（右），由**Bill Pulleyblank**设计

5.5.3 变量取值的上界

此前没有解释过，度约束线性规划的松弛之所以非比寻常，是因为该题目必然存在所有变量取值均为0、1或2的最优解，所以我们完全不必担心边取分数值的问题。子回路的松弛并没有这个结论，它的变量赋值有时候也会出现复杂麻烦的分数。最终，我们有必要从整体上处理一般情形，但是有价值的特例能让我们赢在起跑线上。事实上，在度约束线性规划的解中，如果变量x_{ij}取值为2，就意味着对应的子回路只包含城市i和城市j一来一回的两段路途。根据S={i, j}确定的子回路不等式，可以排除这种情形，但是简单规定$x_{ij}≤1$可以更方便地解决这个问题。这样一来，不但使用度约束，而且限制所有变量的取值上界，就比只用度约束的基础效果更好，所以实际应用往往都采取这种做法。另外，在改进之后，初始模型必然存在所有变量取值均为0、1/2或1的最优解。它虽然不完全是整数解，总算也是八九不离十了。

5.6 完美松弛

借助子回路不等式，我们可以将TSP的解限制在最优解附近，那么，能否再添加其他规则，大大改进所得结果呢？没问题，放心吧，能！要进一步改进结果，需要用到少许数学知识，马上你就明白了。

5.6.1 线性规划的几何本质

迄今为止，我们处理线性规划时，一直在对变量和方程进行运算，把它视为单纯的代数问题。对George Dantzig和大多数线性规划研究人员来说，这样的待遇是不公平的。在他们心目中，线性规划独具一种优雅的几何韵味。

下面，从一道小例题出发，我们试着回顾一下之前忽视了什么。

目标：x+2y取最大值

约束条件：

$$x + y \leqslant 13$$

$$x \leqslant 8,\ y \leqslant 8$$

$$x \geqslant 0,\ y \geqslant 0$$

我们可以把该题的可行解理解成二维平面内的点(x, y)，这里x和y的数值分别由水平方向和垂直方向的数轴确定。

在一道线性规划题目中，全部可行解的点构成的集合称为可行域（feasible region）。为了理解可行域的范围，首先只看第一个约束条件$x+y \leq 13$。相应的直线$x+y=13$把所有形如(x, y)的点划分成两个集合，分别位于直线两侧，直线一侧的点都不是可行解，而直线另一侧的点以及直线上的点都是这个约束条件的可行解。可行解所在的一侧区域称为半空间（halfspace）。图5-15给出了半空间的两种不同表示形式，左图中红色箭头指向可行域一侧，右图中可行域则涂成了红色阴影。



图**5-15**　由一个线性不等式确定的半空间

上面的线性规划例题中，有五个线性约束条件，对应五个半空间，它们的交集就是题目的可行域，如图5-16所示。在这个几何模型里，右侧小图中的蓝色直线代表目标函数，线性规划问题的目标就是尽力向上移动蓝线，同时保证它与可行域有公共点。因此，本例题的最优解就是图中的红点，坐标为(5, 8)。

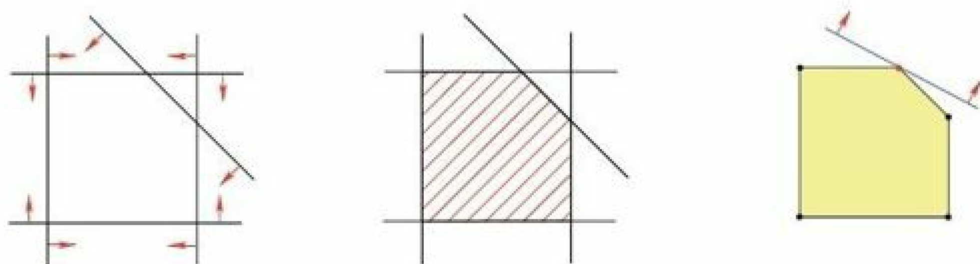


图**5-16**　从几何角度理解线性规划问题

请注意，最优解是可行域的五个顶点之一，另外四个点按顺时针顺序依次为(8, 5)、(8, 0)、(0, 0)和(0, 8)。在这道线性规划题目中，有一个重要结论：无论目标函数如何设定，这五个顶点中必有一个是最优解。事实上，改变目标函数就意味着改变蓝线的斜率，而尽力向某个方向移动直线时，无论移动方向如何，直线在即将离开可行域的瞬间都必定刚好触及可行域的一个顶点。

上述一般性质非常有用，它意味着线性规划问题虽然有无穷组可行解，但求解时只需要考虑顶点即可，而顶点的数目是有限的。事实上，单纯形算法的过程可以理解为沿着可行域边界移动，顺次经过各个顶点。

读到这里，你可能会问：既然只需要把顶点列出来就能解决线性规划问题，为什么还要大费周章地使用单纯形算法呢？原因归根结底在于数字。请记住George Dantzig的话：“对于大规模的问题，组合（顶点）的数目有可能非常多，可能像天上的星星一样多。”在d维空间里，一系列d维半空间的边界面相交，就得到线性规划模型的顶点。这样的交点

未必总会落在可行域里，但是已经足够让试图列出所有顶点的解题者为之头疼了。

5.6.2 闵可夫斯基定理

刚才我们讨论的对象是线性规划问题的顶点，但是具体说到TSP，情况就完全相反了。对于TSP，我们已经有了所有点的完整列表，它们分别对应于图里的各条回路，但是我们却并不知道线性规划模型的可行域是什么样的。从这种角度看来，搜寻TSP规则就等于搜寻能够限制回路范围的半空间。

让我们研究得更仔细一些。在6座城市的TSP里，每条完整路线都对应于15维空间里的一个点，这15维空间里的每个维度都对应于一对城市。每个路线点的坐标都由0和1组成，其中1表示路线用到了相应两座城市之间的边。这些坐标取值或为0或为1的点组成了一个很大的集合，虽然它们都落在15维空间里，但是当我们想选出对应最短路线的点时，却没有任何可供使用的几何结构。线性规划的作用正是弥补这一缺陷。

15维空间的图像是画不出来的，所以我们转而思考二维空间的类似问题，也就是从一组形如(x, y)的点里选出一个，使某个确定的目标函数在该点处取得最大值。这里，我们要找这组点都满足的线性不等式，换言之，使这组点都落在可行一侧的半空间。如图5-17所示，根据实际题目可以逐步构造合适的不等式，最终在给定点集周围得到6个半空间，这些空间的中心区域的每个顶点都是题目条件给定的点。根据线性规划，这个结果具有出色的推论：对所得区域应用单纯形算法，便可以得到原题的最优解。

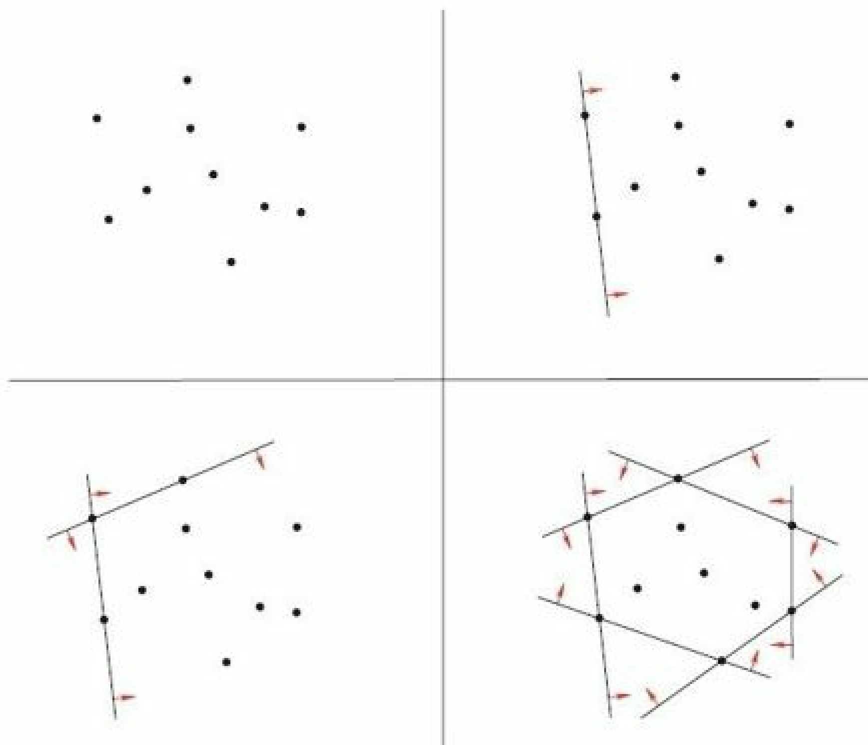


图**5-17**　构造线性约束条件

图5-18再次展示了图5-17最终得到的区域。该区域之所以构造得天衣无缝，并不仅仅出于偶然运气而已。实际上，我们可以通过合理构造，让任意点集都落在线性规划的可行域内，使得可行域的每个顶点都是点集里的一个点。如果是在二维空间里，请想象把一根橡皮筋抻开，然后松手让它收紧，套住所有的点；在三维空间里，请想象用塑料薄膜裹住所有点，然后加热使薄膜缩紧，再次形成完美贴合。如果维度继续增加，那么很难进行形象化的想象，但是结合代数学和几何学，不难证明相应的结论。这个结论最初发现于20世纪伊始，发现者是（赫尔曼·闵可夫斯基）Hermann Minkowski。

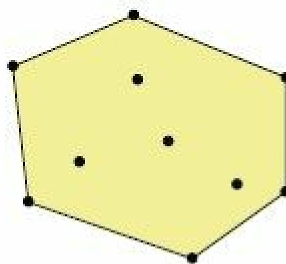


图**5-18**　凸包

给定一个点集，恰好紧紧包围所有点的区域称为点集的凸包

（convex hull）。在二维空间里，凸包是一类特殊的多边形。在三维空间里，凸包则是具有多个小面的立体结构，像是柏拉图立体[1]或者钻石经过切割以后的形状，这些形体统称为凸多面体（convex polytope），数学家从很久以前开始就一直在研究它们。Günter Ziegler在专著*Lectures on Polytopes*（《多面体课程》）中对这一研究领域做出了出色而严谨的介绍，这本书虽然有高深的理论细节，但是引言部分以及前几章内容都值得一读，能让你充分体会为何数学家始终对这些经典几何结构一往情深。[2]

1. 即正多面体，共五种，分别为正四面体、立方体、正八面体、正十二面体和正二十面体。——译者注

2. Ziegler, G. 1995. *Lectures on Polytopes*. Springer, Berlin, Germany.

5.6.3 TSP多面体

任何TSP都可以准确描述为线性规划问题，这个结论非常重要，而闵可夫斯基定理的意义完全不在前者之下！它为寻找TSP规则的行动提供了坚实的理论支持：必需的线性不等式就在那里，只等着我们发现呢。

先别摩拳擦掌，我得提醒你，求解的时候可能会遇到困难，因为描述TSP多面体所需的半空间总数相当庞大。人们已经知道，城市数目达到10座时，所需的不等式数目就至少达到51 043 900 866个。[1]不过，只是数字很大的话，并不足为惧。2008年，Harold Kuhn做了一场George Dantzig纪念讲座（George Dantzig Memorial Lecture），他在演讲里强调了这一点，并在文中援引了自己在1953年的TSP研究做例子。

1. Christof, T., G. Reinelt. 2001. Int. J. Comput. Geom. Appl. 11, 423–437.

> 整个夏天，我跟George经常联系，一直在讨论这道题目和其他题目。我知道，夏天快结束时，他来听过我的课（Selmer Johnson、Ray Fulkerson和Alan Hoffman也来了）。我们俩都很清楚，虽然在用线性规划模型描述旅行商问题的时候，面（face，约束条件）的总数是巨大的，但是只要你能找到一道松弛问题的最优解，这道题由面的子集构成，而且这个解是一条回路，那么你就解决了这一切背后的旅行商问题。

我们需要好好学习如何使用不等式，也就是如何提出对于求解的TSP题目有用处的不等式。这是线性规划求解TSP的核心所在，第6章将

对此进行深入讨论。

5.7 整数规划

如果掌管线性规划的精灵出现，允许全世界的线性规划方法使用者许一个愿望，他们肯定会异口同声地祈求精灵赐给他们整数解。有一个原因显而易见，就是希望避免麻烦，比如要求生产33+1/3个产品的生产方案就相当棘手。但是，最主要的原因还是为了把决策归结为一系列独立的选择。是否应该建造一座新工厂？请回答“是”或者“不是”。是否应该生产销售一种新产品？请回答“是”或者“不是”。如果可以引入只能取值0或1、不能取分数或小数值的变量，那么线性规划模型就能给出这样的决策。这是线性规划的一类推广，它虽然威力强大，但是现阶段却离不开巨大的计算成本。

对变量附加整数限制之后，线性规划模型就不符合Dantzig的理论了，因此无法由单纯形算法或其他线性规划方法直接求解。尽管如此，成千上万人无法抗拒整数变量带来的灵活性，所以照样迎难而上，每天都在线性规划模型中使用这样的限制。这种拓展的模型称为整数规划（integer programming，简称IP）。

Dantzig最早做出了关于整数规划广泛适用性的书面记述。他的一篇论文同时为最优化理论和复杂性理论奠定了基础，其中也说明了如何将一长串重要的最优化问题逐一建模转化为整数规划问题。[1]在1963年的线性规划专著中，他对这番研究工作作了如下的描述[2]：

1. Dantzig, G. B. 1960. Econometrica 28, 30–44.
2. Dantzig (1963).

> 我们的整体目标是对一些题目进行总结并分类，这些题能够归结为线性规划问题，而且其部分或者全部变量只取整数值。我们将说明，大量非线性、非凸性的组合问题虽然复杂困难，乃至表面看来完全无从下手，如今都有机会直接正面攻破。

他的题目分类包括TSP和最优地图染色问题。

染色问题是整数规划的典范应用实例。我们不考虑具体的地图，而是考察更一般的问题：对图的顶点染色，使得任意两个共边的顶点都不同色。我们在地图的任一区域内放入一个顶点，如果两个区域有公共边界，就在对应的两个顶点间连一条边，便可以把地图染色问题转化为上述一般的染色问题。

图5-19给出了一个用4种颜色染色的图。对于这一特例，不难看出，

无法只用3种颜色给出符合要求的染色方案，但是一般而言，我们很难确定3种颜色够不够用。下面就把这种一般问题转化为整数规划问题。每个顶点i的颜色用3个非负变量$x_{i,red}$、$x_{i,green}$和$x_{i,blue}$表示。如果顶点i染成红色，那么令$x_{i,red}$取1，反之则取0，$x_{i,green}$和$x_{i,blue}$则分别对应绿色和蓝色，同样在0和1中间取值。因为顶点i的颜色必须指定为红绿蓝之一，所以有约束条件

$$x_{i,red}+x_{i,green}+x_{i,blue}=1$$

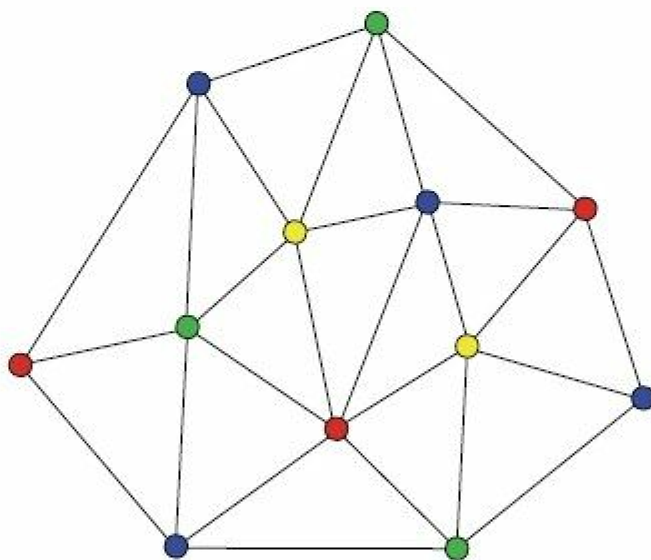


图**5-19** 用**4**种颜色染色

对于任意边（i, j），因为两端的顶点i和j不可能染成同一种颜色，所以有约束条件

$$x_{i,red}+x_{j,red}\leqslant 1$$

$$x_{i,green}+x_{j,green}\leqslant 1$$

$$x_{i,blue}+x_{j,blue}\leqslant 1$$

上面这组方程和不等式的整数解给出切实可行的红绿蓝染色方案。不过，如果不限制变量取值为整数，则令所有变量等于1/3也是满足条件的线性规划解，请注意，这个解完全不能提供任何对染色有用的信息。一切的意义都取决于整数变量。

5.7.1 TSP的整数规划模型

在某种意义上，我们已经成功建模，将TSP也视为一类整数规划问题。事实上，有了整数变量，再把度约束条件和子回路不等式结合起来，已经足够保证TSP题目的任何线性规划解都必定是一条完整回路。这个数学模型虽然属于整数规划模型，但并不能直接交给解题程序让它解决。问题在于，如果TSP题目包含*n*座城市，那么它的子回路不等式总数就近似等于$2^{n/2}$。

因此，Dantzig采用了另一种模型描述TSP题目，这次他使用了n^3个非负变量，但约束条件只有$n+n^2$个。模型的核心思想是，对于两个城市间的边，我们不仅标明它是否用在路线中，同时也标明它在路线中出现的顺序位置。Albert Tucker等其他早期TSP研究者曾经用过其他整数规划模型，那些模型变量和约束条件比这个模型更少，但是它们并不够切实可行，对旅行商用途不大，我们不打算耗费笔墨。迄今为止，所有其他模型在实际应用性能方面都不及子回路模型，也不如我们将在下一章介绍的方法。看到这些复杂的模型，你应该能够明白，要想从整体上解决整数规划问题很困难，因为多项式时间的整数规划解题程序就直接意味着多项式时间的TSP解题程序。

5.7.2 整数规划的求解程序

虽然解决一般性的整数规划问题是个难题，但世人并未因此却步，依然在海量商业应用中反复建立整数规划模型。就像面对TSP时一样，假如我们举手投降，向世界宣布，一般的整数规划问题也许压根不可能解决，那根本无济于事。我们必须用算法工程的方法攻克它。事实上，有人已经做到了，而这方面的成果大致就是TSP研究者的主要回报：几乎所有已知的整数规划问题通用解法，最初都是在求解TSP的过程中提出的。

至于全世界建立的商业整数规划模型，目前市面上有几种精巧复杂的求解程序正处在竞争之中。在过去20年间，这些计算机程序的实际性能获得了有力的提升，而放眼未来，我们会进一步了解如何打败旅行商（TSP）和整数规划（IP）这对兄弟，因此它们的求解程序必将迎来更大程度的改进。

5.8 运筹学

线性规划和整数规划的研究人员遍布数学领域、计算领域、商业界、科学界及工程界。然而，与它们关系最为密切的学科却有个独特的名字：运筹学（operations research，简称OR或O.R.）。

回溯运筹学的历史，它植根于20世纪最初几年的军事规划，由“运筹帷幄决胜千里”的宗旨而得名“运筹”。现在，运筹学领域的科研系所和研究中心遍及全球。在美国，有运筹学教学项目的院校包括加州大学伯克利分校、卡耐基梅隆大学、哥伦比亚大学、康奈尔大学、佛罗里达大学、佐治亚理工学院、利哈伊大学、密歇根大学、麻省理工学院、西北大学、普林斯顿大学、罗格斯大学、斯坦福大学，等等。

市场营销机构可能不会提议把这门学科命名为运筹学。事实上，营销人员为此提出了一句口号：“关于更好的科学（The Science of Better）。”这是一场市场推广活动的核心口号，活动发起方是一家专业学会，全名为运筹学和管理学研究协会（Institute for Operations Research and Management Science），简称为INFORMS。对于“什么是运筹学？”这个问题，该学会做出了如下回答：“总而言之，运筹学是关于应用高等分析方法帮助人们做出更好的决策的学科。”总结得很好，正中要点：运筹学方法是放之四海而皆准的通用方法，从医疗保健到交通运输，从经济财政到林业生态，只要需要决策，运筹学就可以派上用场。在运筹学研究中，线性规划、整数规划等种种最优化工具与借鉴自概率论、博弈论以及其他领域的建模方法相辅相成，联合发挥威力。

要想感受运筹学的热度与广度，最好的切入点就是Michael Trick的文章。他是卡耐基梅隆大学的教授，曾经担任INFORMS的主席。他的博客更新频繁，关于运筹学的内容应有尽有，无所不包，自然也包括他的专攻方向：运筹学在体育规划方面的应用。[1]他独具慧眼，总能发现最优化方法的实际应用机会。所以，请密切关注他的博客，说不定会看到TSP的新应用呢。[2]

1. 参见http://mat.tepper.cmu.edu/blog/。

2. Laura McLay的“朋克摇滚运筹学”博客（http://punkrockor.wordpress.com/）也很不错，同样适合关注运筹学进展的读者。



图**5-20　Michael Trick，2010**年

第6章 割平面法

> 首先，Dantzig、Fulkerson和Johnson用绳子在实体模型上得到一个解（实际上也是最优解），接下来，他们依然必须面对无数种可能的切割方式。
>
> ——Alan Hoffman和Philip Wolfe，1985年[1]

1. Hoffman和Wolfe (1985).

与旅行商问题相关的线性规划的松弛问题极为复杂，因为单纯形算法无法解决有几十亿约束条件的大型题目。幸运的是，Dantzig、Fulkerson和Johnson面对这种复杂性，提出了一种巧妙的处理方案。他们的方法称为割平面法。割平面法的目标不是一举解决整个线性规划问题，而是以“随用随取”为原则，只在必要的时候才产生相应的TSP不等式，从而逐步计算线性规划的边界。从此，游戏局面改变了，而且这种改变并不局限在旅行商的世界里。

6.1 割平面法

Dantzig等人得到了环游美国的路线，而他们上路的第一步就是度约束线性规划的松弛问题及其解，如图6-1所示。图中，红色的边取值为1/2，黑色的边取值为1，而其他变量取值均为0。

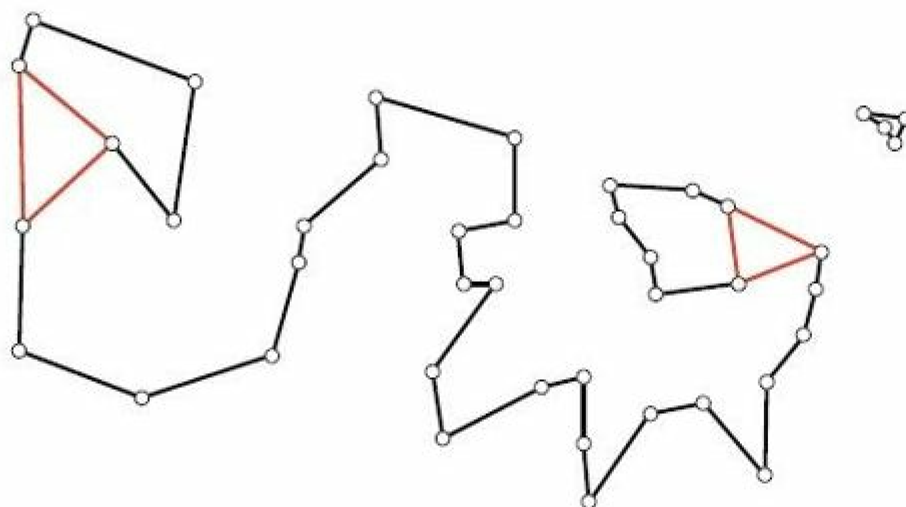


图**6-1**　变量取值有上界时，度约束线性规划的松弛问题的解

一看即知，单纯形算法确实没错：与每座城市相连的各条边取值之和都是2，也就是说，要么是两条黑边，要么是两条红边加一条黑边。此外，图6-1的解显然不是一条完整回路。最明显之处在东北角，那里有四座城市独立成环，孤立的子回路形成了一个孤岛，而锱铢必较的旅行商不乐意选择任何一条从孤岛通往外界的路。如果我们能够一五一十地列出所有子回路消去约束条件，那么这个问题就不复存在了。但是列出这些条件也就意味着，要向线性规划模型中加入2 199 023 254 648个不等式，这给线性规划求解程序提出了一个超级棘手的大难题。

毋庸置疑，兰德公司研究小组有优秀的人工计算能力，但是显然他们当初并没有拿到两万亿的子回路不等式直接计算，而是选择了巧妙得多的迂回解法。他们以图6-1的线性规划解为指导，着手寻找对所有路线都成立的不等式，而且不等式数目不多不少，刚好够用，从而保证单纯形算法返回的最优解就是一条完整回路。因为所有周游各城市的回路都是该线性规划模型的可能解，所以单纯形算法给出的那条回路肯定不但是最优解，也是最优回路。

下面开始解题，步骤如图6-2所示。第一步是解决东北四城市孤岛的

问题，为此选择一个相应的子回路消去约束条件，如第一幅小图所示。我们把这个不等式添加到线性规划模型里之后，单纯形算法给出一个新的解，如第二幅小图所示。这时，在美国东北五大湖地区出现了另一个孤岛，由七座城市组成，于是我们再加入另一个相应的子回路不等式。单纯形算法又给出新的解，如第三幅小图所示。可是，这次的解里依然包含一个新的四城市孤岛，位于美国中部。这种做法看起来就像是修补年久失修的堤坝，东边刚刚堵上一道决口，西边立刻就出现一道新的。事实上，Dantzig等人的论文初稿里提到，研究者Edwin Paxson称该方法为“手指堵堤坝解法”[1]，他是该小组在兰德公司的同事。

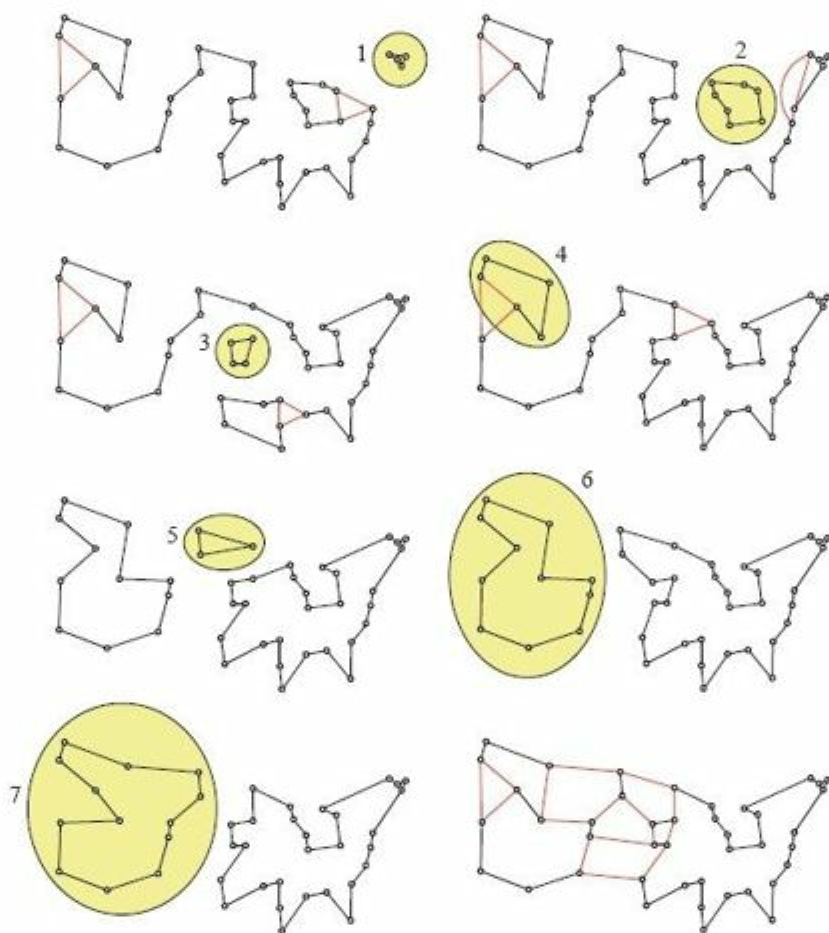


图**6-2**　割平面法的前七步

1. Dantzig, G., R. Fulkerson, S. Johnson. 1954. Technical Report P-510, RAND Corporation, Santa Monica, California. 典故出自经典儿童文学作品《银冰鞋》，作者是美国著名儿童文学作家玛丽·梅普斯·道奇。“手指堵堤坝”的故事只是书中虚构的小插曲，后来在美国家喻户晓，其主要内容是一个“勇敢”的荷兰小男孩发现堤坝有个小口子，便用手指堵了整整一夜。——译者注

然而，外表并不是真相。诚然，他们的解法看起来可能并没有缩短与所求回路之间的距离，但是实际上确实在大步前进。按照最开始提出

的线性规划模型，最优解对应的目标函数值为641，加入第一个不等式用来消去东北角的子回路之后，相应的目标函数最优值就提高到了676，而加入第二个不等式之后，更是把这个值提高到了681。最优解对应的目标函数值是最优路线长度的下界，所以虽然每次都会冒出讨厌的孤岛，但我们的方向显然没有走错。接下来的五步如图6-2所示，最优路线长度下界依次改进为682.5、686、686、688、697。

上面的最后一步给出了697的下界，而实际的最优路线长度是699，两者之差只有2个单位长度而已。胜利的曙光真是太棒了！可是要如何继续呢？长度为697的解已经不包含任何孤岛了，但是没有孤岛并不是问题所在。仔细回顾之前几步就会发现，第四步的解也是连通的，没有孤岛，当时的问题出现在西北角上，一组城市构成的团簇与外界其他城市只通过两条红色边相连，取值之和为1。再回到最后一步的解，我们并不能明显看到这样的城市集合。事实上，之后会说明，最后一步的解其实满足所有的子回路消去约束条件。这个结论很有意思，因为这意味着我们只加入了7个不等式，就计算出了满足所有子回路消去约束条件的最优解，而这些不等式的总数超过两万亿！确实是很有意思，可惜不足以解决这道42座城市的TSP题目。我们还是得想法设法把下界提高到最优路线的长度才行，也就是要从697提高到699。

好吧，所有子回路消去约束条件统统没用了。没关系，在描述这道题目的TSP多面体中，还有好多好多别的不等式可以选择呢。我们只需要找到上面的解不满足哪一个就行了。Dantzig等人当时借助富有新意的专门论证，明确描述了两个合适的不等式。我们可以不必使用专门论证，而是给出一条一般性的规则。这条规则讨论的是所给城市的4个子集，它们之间的关系如图6-3中左图的文氏图所示。自己试着连出几条回路，比如右图展示的那一条，你就会明白，每条回路穿过4个集合边界的次数必然是至少10次。换言之，把这4个集合对应的子回路不等式加起来，再把不等式右端换成10，则每条回路都必须满足这个新的约束条件。[1]

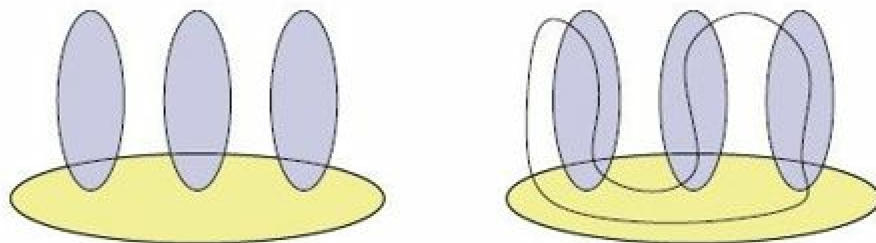


图**6-3**　四集合结构，此时回路穿过集合边界的次数必然是至少**10**

次

2. 请注意，满足子回路不等式取值为2的回路只有一种情况，就是进入该子集，到达其中每座城市，然后离开该子集。因此，在该子集内部，这条路线相当于一条简单通路。如果图中3个蓝色集合都有这么一条通路，那么黄色集合就必须与路线相交至少3次。但是一条路线与任一集合边界都必然相交偶数次，也就是说黄色集合与路线必须相交至少4次。综上所述，路线与3个集合各相交2次，与1个集合相交4次，因此相交的总次数至少为10。

有了上述的四集合结构，环游美国问题的最后几步也可以完成了，如图6-4所示。第一幅图标出了4个集合，它们对应的不等式就是第8个约束条件。黄色集合边界共被3条黑边穿过，其中1个蓝色集合边界由4条红边穿过，另外2个蓝色集合边界则都由2条红边和1条黑边穿过。把这些数目加起来可知，穿过这4个集合全体边界的黑边总数是5条，红边总数是8条，在线性规划模型中，取值之和为9。[3]之前我们要求这个数值应该至少为10，所以这就是我们要找的不等式。把相应的四集合不等式添加到现有的线性规划模型中，重新求解，得到的最优解取值为698，如图6-4的第二幅小图所示。再加入另一个四集合不等式，终于大功告成了：单纯形算法给出了一条新路线。现在，借助对偶线性规划问题的解，我们可以让所有人相信，699确确实实是这道TSP例题的最优值。

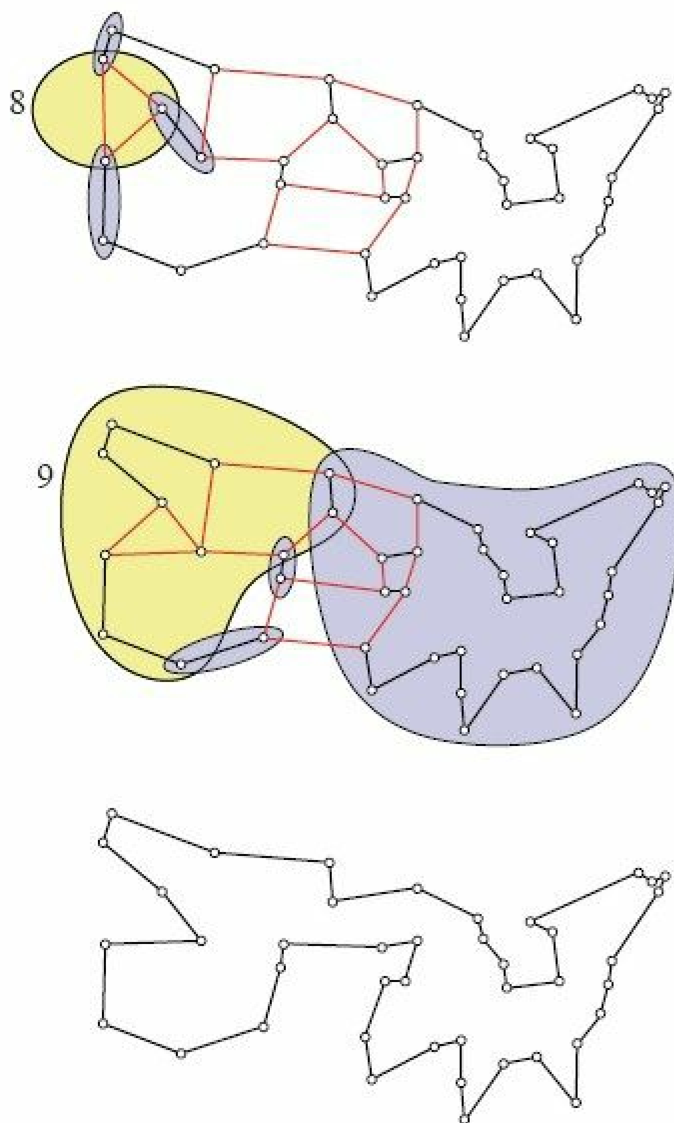


图**6-4**　割平面法的第八步和第九步

3. 注意在红边三角形中，每条红边都分别穿过两条边界，因此计算总数时就要数两次。

我们总共用了9个不等式，解决了旅行商周游美国的路线难题。所有不等式共有两万亿个，实际解题时却只用到了9个，这真是太神奇了！兰德公司研究小组慧眼如炬，居然敢于尝试这样的逐步求解过程。你如果能意识到所需手工计算的工作量有多大，肯定会为他们的远见卓识所折服。

最终选出的9个不等式称为割平面（cutting plane），因为每一步，对应的半空间（也就是半平面）都把线性规划模型的可行域一割两半，使当前的解落在切掉的那一块里。“割平面法”这个名字不像Paxson提议的“手指堵堤坝解法”那样引经据典、辞藻华丽，但是也没有那么悲观绝

望。

在那篇著名论文的结尾处，Dantzig等人谦逊地写下了下面这段结语。[4]

4. Dantzig et al. (1954).

> 显然，对于人们可能提出的理论本质与旅行商问题相关的任何题目，我们几乎都没有给出解答。但是，我们希望本文已经有效地展示出了，包含的点不多的题目是可以得到解决的，也希望这里用到的部分思想同样能够用在具有类似本质的其他问题上。

确实很有效！他们的研究工作影响深远，余韵至今仍然在应用数学界回荡不息。

6.2 TSP不等式一览

Dantzig等人在解题过程中用到了两个非子回路不等式，也就是说这两个不等式并不属于子回路不等式。其中，第一个不等式对应前面讲到的第一个四集合结构，只在形式上有所变化。但是第二个不等式却与之截然不同，原始论文脚注表明，Irving Glicksberg是幕后的无名英雄："我们感激兰德公司的I. Glicksberg，他向我们指出了这一类的关系。"[1]

1. Dantzig et al. (1954).

Fulkerson很清楚，最好的做法也许是不要依赖特定的具体论证，就算提出论证的人是他们的朋友Glicksberg，也一样不能依赖。因此，他向TSP研究专家Isidor Heller写了一封信，信上签署的日期是1954年3月11日。他用符号"Cn"表示一道题目全体路线集合的凸包，题目包含的城市数目为*n*。

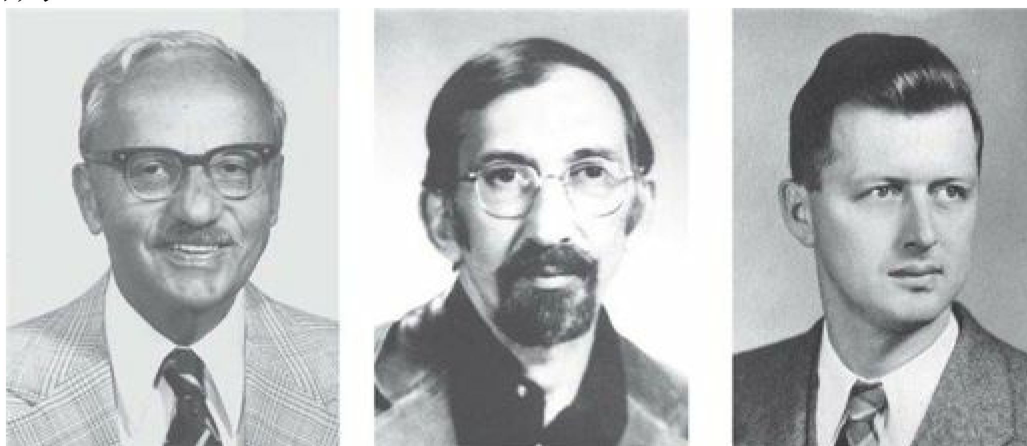


图**6-5　George Dantzig**（左）、**Ray Fulkerson**（中）和**Selmer Johnson**（右）。（照片分别由美国国家工程院、**Merle Fulkerson Guthrie**和得克萨斯大学美国历史中心提供）

> 最近，我和G. Dantzig、S. Johnson一直在借助线性规划方法研究该问题的计算特性，尽管我们并不知道，对于一般的n，路线的凸包C_n的各个面究竟如何。不过，我们使用的方法看来有望获得成功，特别是对于一道48座城市的大规模题目，我们已经通过手工计算相当快速地找到了最优路线。我们发现，根据各点所在的地图，可以方便地利用该方法对Dantzig的单纯形算法进行转换，从而确定方向相反、其他都相同的路线。例如，可以用10维空间中的25个超平面构成的系统描述C_5。我们还不太清楚C_n的一般性质，但是如果有机会读到你的

> 论文，也许我们会学到更多。

Dantzig也向Harold Kuhn写信提出了类似的请求，时间是同一天，即1954年3月11日；他还向Albert Tucker写信求助，时间是1954年3月25日。显然，兰德公司的这几位研究者当时都在积极寻求关于TSP多面体结构的知识，以便加强割平面法的能力。

6.2.1 梳子不等式

尽管Dantzig等人发出了求助信件，但是研究界的反馈速度很慢。那个年代的科技没有现在这么发达，确实存在落后不便之处。20世纪70年代初，Vašek Chvátal终于开始重新研究这个主题。[1]他得到了关于梳子不等式（comb inequality）的研究成果，推动了这方面的研究再次起步，而这距离兰德公司研究小组的原始工作已经过了接近20年光阴。Martin Grötschel和Manfred Padberg紧随其后，推广并分析了梳子不等式，为后续研究提供了效仿的样板。[2]



图**6-6 Vašek Chvátal**（照片由**Adrian Bondy**提供，所有权利保留）

1. Chvátal, V. 1973. Math. Program. 5, 29–40.

2. Grötschel, M., M. W. Padberg. 1979. Math. Program. 16, 265–280.

这里，所谓的梳子其实指的是由一系列顶点构成的一组子集，它们的关系如图6-7文氏图所示。黄色集合称为“梳柄”，蓝色集合则称为“梳

齿”，梳齿之间互不相交，梳柄与每个梳齿都相交（交集非空）。我们要求，梳齿的数目*k*应该是不小于3的奇数。可以证明，每条回路穿过梳子各边界的次数必然至少是3k+1次，从而推广四集合结构的结论。不过证明时需要细心周密，避免遗漏某些可能的情形。

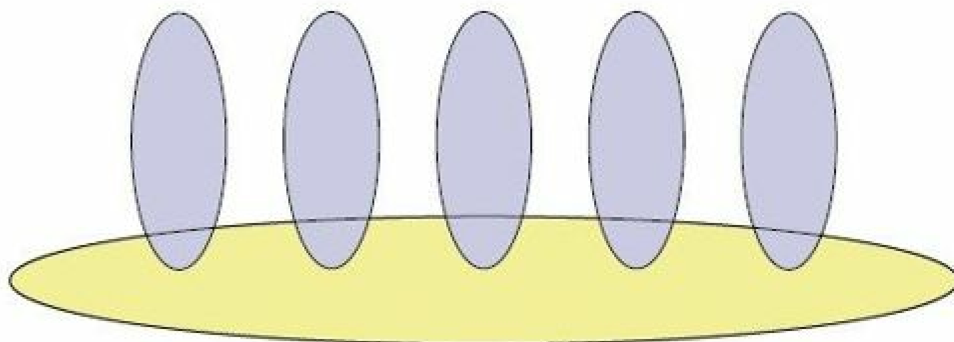


图**6-7**　一把有**5**个梳齿的梳子的文氏图

如何理解次数至少是3k+1次？仔细观察图6-8，数一数就知道了。第一幅图的梳子有5个梳齿，一条路线遍历所有顶点，与边界的交叉次数是16，也就是3×5+1。第二幅图的梳子则有6个梳齿，一条路线遍历各顶点，但是这幅图是错误的，我们不允许出现这种情况，因为6不是奇数，路线与边界的交叉次数也只有18，比3k+1规则少1，不满足要求。在梳齿为偶数的情形下，梳齿可以两两配对，不需要像奇数情形那样必须增加一次交叉才能穿出梳柄（即第一幅图路线与梳子最右侧的交叉点）。

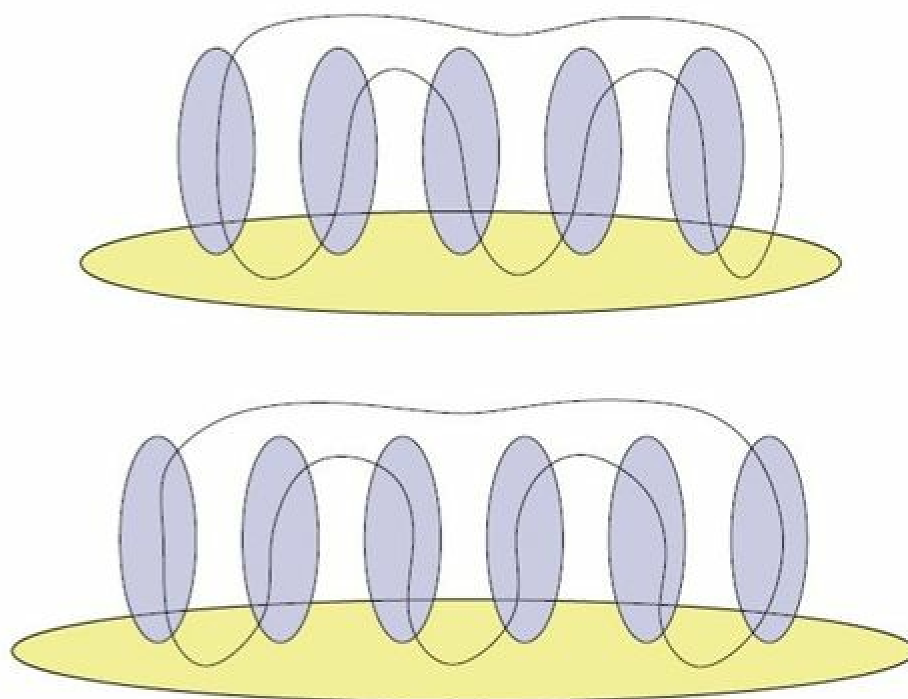


图**6-8**　路线穿过梳子，上图梳子有**5**个梳齿，下图有**6**个梳齿

我们已经知道，42座城市的环游美国之旅之所以能圆满完成，就是梳子的功劳。Grötschel也用梳子模型解决了另一道题目，给出了环游德国120座城市的最优路线，创下了TSP解题的新纪录。图6-9是他在解题时手工绘制的，图上不仅画出了线性规划的一个解，也表明了下一步可能用到的割平面。我们有幸欣赏这幅真实的原作：取值为1/2的边画成了波浪线，不满足子回路不等式的区域用红笔圈出，不满足梳子不等式的区域则由蓝笔圈出相应的梳柄和梳齿。每经过一轮计算，他都会向线性规划模型里增加好几个割平面——遇到大规模的旅行商问题，就应该采取这种做法，提高求解效率。

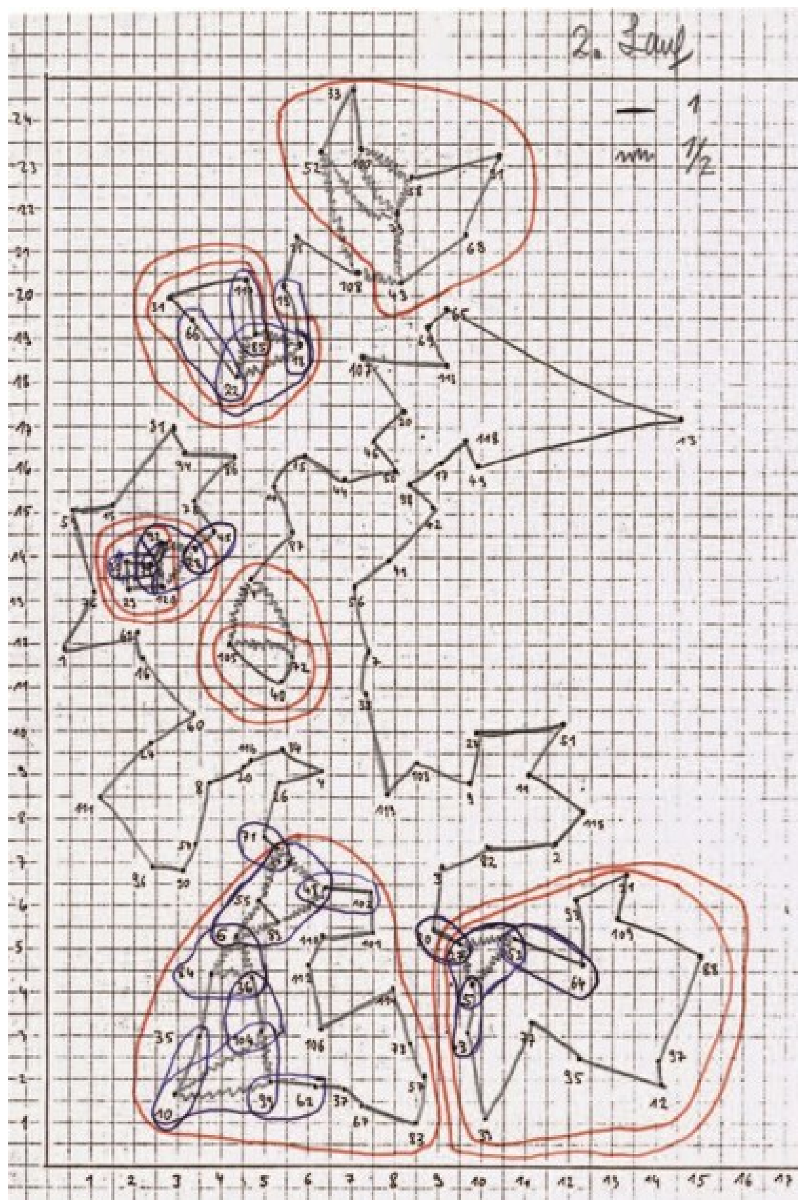


图**6-9　Martin Grötschel**的割平面图示，**Martin Grötschel**供图

6.2.2 TSP多面体的小平面定义不等式

Grötschel在计算上取得了成功，唤醒了研究界对其他类型TSP不等式的研究热情。此后，他本人与William Pulleyblank合作，率先得出了新成果，把梳子结构模型由一个梳柄的情形推广到多个柄的情形。[1]多个柄的新结构称为团树（clique tree），因为满足要求的文氏图形状与树有几分相似，图6-10就是一例。

1. Grötschel, M., W. R. Pulleyblank. 1986. Math. Oper. Res. 11, 537–569.

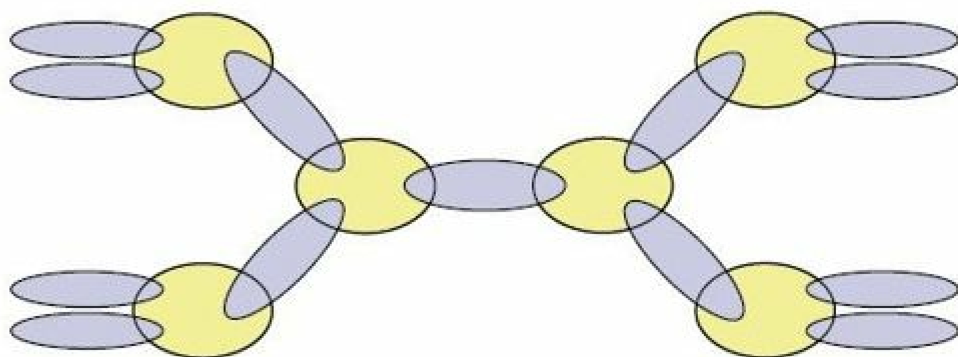


图**6-10**　团树

在Grötschel和Pulleyblank的研究之后，其他研究团队纷纷得到了形状比团树还要奇妙纷繁的结构。当然，奇妙纷繁只是那些不等式的表象，研究工作本身则始终没有偏离TSP多面体基本结构的路线。在高维TSP世界里，很难描述清楚TSP多面体这种几何形体，但是退回到二维平面，思路就豁然开朗了。

请考虑图6-11所示的凸包。图中画出了两个半空间，虽然它们都与集合中的至少一点相切，但是毫无异议，上面的半空间比下面的更具有本质性。事实上，在这个例子中，上面这类半空间总共只有6个，而它们合起来就能给出凸包的完整描述。这6个半空间称为小平面定义不等式（facet-defining inequality），它们围成的6条边则称为TSP多面体的小平面（facet）。

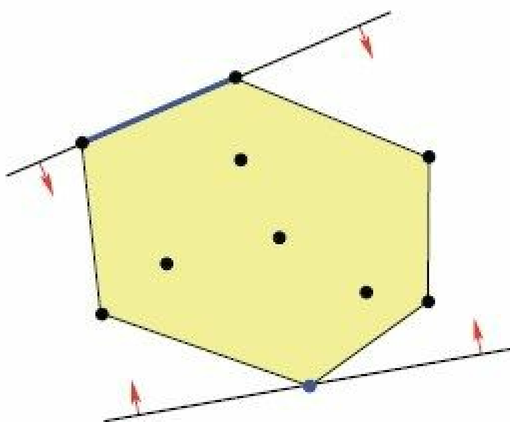


图**6-11**　小平面定义不等式和非小平面定义不等式

我们虽然无法画出TSP多面体的准确图象，但是这里定义的小平面概念却同样适用于高维情形。[2]对于TSP路线的凸包，小平面定义不等式与度约束条件可以共同给出完整描述，而任何完整描述中，这两类不等式也必然缺一不可。正因如此，我们尽管不清楚TSP多面体的完整结

构，依然可以得出肯定的结论：在任何一组能够恰好紧紧包围住路线集合的不等式之中，必然存在某些特定的公共不等式。

2. 为证明一个不等式是小平面定义不等式，可对满足不等式的回路方程进行分析。选取包含这些路线对应点的最小平面，验证它所处的空间维度恰好比TSP多面体空间维度小1。二维空间的例子便是如此，小平面上的两个顶点决定一条直线，而直线也就是一维平面。

20世纪50年代，Harold Kuhn和其他人都研究过小TSP多面体的小平面。但首先倡议研究界关注小平面定义不等式的并不是他们，而是Grötschel和Padberg。[3]这两人当时正在构造有可能性的割平面列表。

3. Grötschel, M., M. W. Padberg. 1979. Math. Program. 16, 265-280.

> 我们对证明该事实的兴趣基于两个方面：首先，在提出的这些不等式中，哪些对于定义这个极为复杂的多面体有真正重要的意义，获知这个问题的答案是富有数学趣味的；其次，按整数规划的意义理解，小平面是“最强的割平面”，因此我们自然会猜想，这些不等式对于获得这道组合最优化难题的数值解也具有至关重要的计算意义。

Grötschel和Padberg共同证明了子回路不等式和梳子不等式都是小平面定义不等式，而Grötschel又和Pulleyblank一起证明了团树不等式也属于这一类“精英不等式”。

整个20世纪90年代，小平面一直是研究的热点话题，领军人物是法国格勒诺布尔的Denis Naddef和意大利罗马的Giovanni Rinaldi。[4]这段时期的研究成果非常丰富，大量信息至今尚未在计算科学领域得到充分利用。尽管如此，我们对于TSP多面体的了解仍然非常有限，距离彻底认识仍有很大差距：10座城市的多面体就已经发现了足足510亿个小平面，而现有的一般性理论只能解释其中的一小部分。说得乐观一点，该领域仍有大量信息等待后来者发现，为未来的TSP研究提供了充足的研究目标和发展空间。

4. Naddef, D. 2002. In: G. Gutin, A. Punnen, eds. *The Traveling Salesman Problem and Its Variations*. Kluwer, Boston, Massachusetts. 29–116.

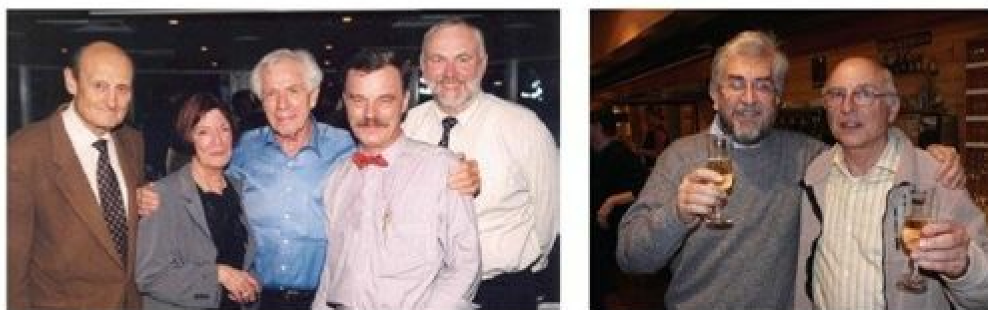


图**6-12**　左图：**Egon Balas**、**Suzy Mouchet-Padberg**、**Harold**

Kuhn、**Manfred Padberg**和**Martin Grötschel**（**2001**年摄于德国柏林，照片由**Martin Grötschel**提供）；右图：**Giovanni Rinaldi**和**Denis Naddef**（**2008**年摄于法国**Aussois**，照片由**Uwe Zimmermann**提供）

6.3 TSP不等式的分离问题

现在我们面前有一大堆不等式，任何有意接受TSP挑战的勇士都可从中自由选用，不过，高效利用这些不等式可没那么容易。事实上，如果要继续推进TSP研究，这个问题就应当重视。

我们的研究任务是分离问题（separation problem），也就是说，要从已知的TSP不等式中，找到给定的线性规划解不满足的不等式。可以认为，这些不等式对应的半平面将给定的解从路线的凸包中“分离”出来，该问题正是因此而得名。分离是割平面法的核心问题——Concorde之类的TSP求解程序的实质只是调用分离程序的打包程序而已。所以，假如你希望加入TSP的盛会，那么研究又快又省的分离方法绝对是不二法门。

6.3.1 最大流与最小割

在割平面法中，各种子回路消去约束条件犹如工蜂，任劳任怨，尽职尽责。至少，在子回路消去约束层面，分离问题已经得到了充分的认识。这里用到的技巧方法可追溯至冷战年代，当时有两拨数学家关注过东欧铁路系统，一方的研究内容是利用铁路运输装备，另一方的研究方向则是如何“卓有成效”地发起轰炸从而摧毁铁路网。[1]

1. Schrijver, A. 2002. Math. Program. 91, 437–445.

我们不打算就此转向欧洲，还是继续看42座城市的环游美国TSP吧。请再次仔细考察加入前7个割平面之后得到的线性规划解。图6-13给出了对应的线性规划图解，我们用四种不同颜色指明了四条通路，它们均从南方的Phoenix出发，汇集于北部的Montpellier。设想要沿着这四条道路运送货物，比如运送油品，而经过每条边的载货量不得超过这条边的x_{ij}值，也就是说用x_{ij}值来衡量连接i点和j点的管道容量。设每条通路赋值为1/2，从源点（出发地）流向汇点（目的地）的总流量就是2。

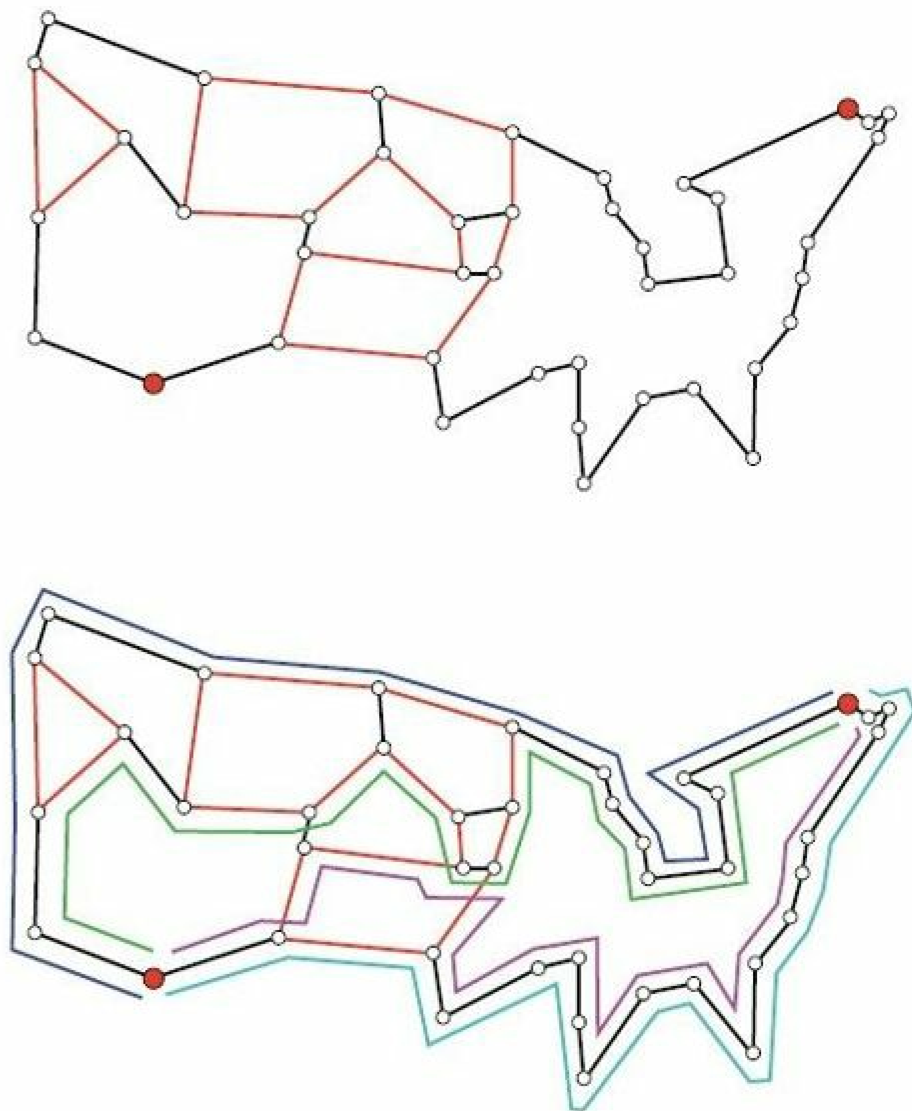


图**6-13　Phoenix**和**Montpelier**之间的**4**条通路

“流”的对偶概念是“割”。割就是一组边，假如在图中去除这些边，图就会变成两个互不相交的独立子图，其中一个子图包含源点，另一个子图则包含汇点，因此一个割把顶点分成两个集合。割包含的各边的容量之和是最大流量的一个上界，因为在运输油品途中，必然会在某时某处离开一个子图，到达另一个子图。在42座城市的例题中，我们可以把一端是Phoenix的所有边取作一个割，它的容量是2，与之前得到的流量相等。这并不是巧合，而是关于流和割的核心定理——最大流最小割定理。

最大流最小割定理是一个具有“强对偶性”的结论，内容是说，对于任意一个图，在图中任意选定源点（source）和汇点（destination），则

流的最大值等于割的最小容量。由此可进一步推出，如果一个标准的多项式时间算法能计算最大流，那么它也能计算最小割。

注意，由割产生的两个子图中，有一个子图对应的子回路不等式取值恰好等于割的容量。又由于从Phoenix到Montpelier的流量为2，可以推知，任取一个只包含这两座城市之一的集合，它对应的子回路不等式都已经得到满足。依次把汇点（目的地）换成剩下的40座城市，如法炮制，便可证明，所有的子回路不等式均已得到满足。[2]

2. 集合S与其补集有相同的子回路不等式，S的补集由所有不在S中的顶点构成。因此，我们可以假定Phoenix属于集合S。

总而言之，我们解决子回路分离问题的一般方法如下：首先选择一个源点，依次以其他顶点为汇点，计算出最大流，也就是求解*n*-1道最大流问题。如果计算发现所有的最大流量都是2，那么所有子回路不等式就都已经得到满足；反之，只要有任何一个结果小于2，则对应的最小割就不能满足某个子回路不等式。[3]听起来求解过程似乎耗时很长，但实际上完全可以实现很快的执行速度。

3. 流量问题是网络理论的基础，自冷战时期以来，网络理论一直在持续发展。欲知详情，可参考《网络流》一书（R. Ahuja, T. Magnanti, J. Orlin. *Network Flows*. 1983. Prentice Hall, Englewood Cliffs, New Jersey.）。

6.3.2 梳子分离问题

子回路消去约束怎么样？没问题，已经搞定了。那么梳子不等式呢？不是太顺利。目前我们还没有找到合适的多项式时间算法，保证能够发现给定解不满足的梳子不等式。我们也没有把梳子分离问题归入NP完全问题的复杂度类，它的地位还没有定论。该问题悬而未决，而又亟需解决，换言之，它是个重要的研究课题。

虽然梳子分离问题尚未解决，但这不代表计算时就完全不考虑梳子。如果子回路消去约束条件不能改进线性规划最优解的界，计算机程序别无他法，必须找出有用的梳子，为此只能不择手段，使出浑身解数。程序做到了，使用的方法是试探策略，能够实现延长梳柄、缩减梳齿、切分集合等诸多操作。当然可能会遇到复杂混乱的情形，但其实在初始阶段很容易发现不满足的梳子结构。事实上，梳子不等式的结构表明，我们在寻找可能的梳齿时，应当观察现有线性规划解，考虑边界交叉次数取值接近2的集合。这样的集合其实有现成的，它们形如$S=\{i, j\}$，这里i和j是解里满足$x_{ij}=1$的值，也就是图里黑色各边的一对顶点。

因此，我们解决这一问题时，第一步可以在图中删掉所有黑色边，考察剩下的红色独立子图。如果这些黑色边与其中一个子图边界的交叉次数为奇数，就确定了一处不满足梳子不等式的结构。求解过程如图6-14所示，这里我们在子回路线性规划的松弛解中找到了两个可能有用的梳子不等式，并在下面两幅图里标明了梳子结构。在计算时，我们选择了左下图中有3个梳齿的梳子。

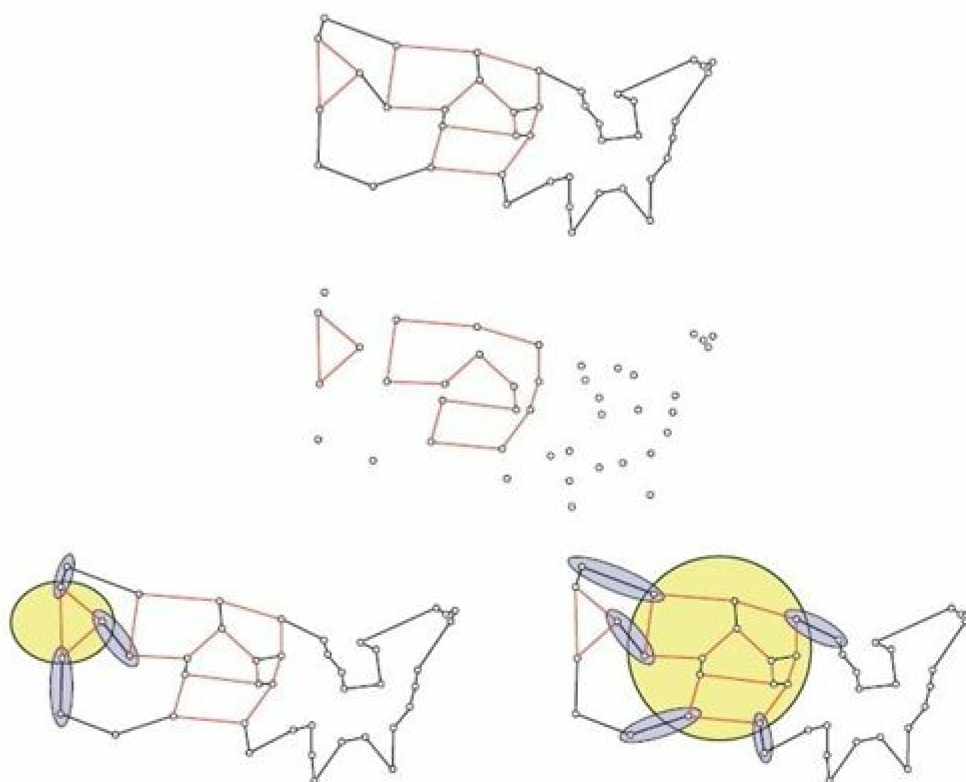


图**6-14**　以红色独立子图为梳柄的两个梳子结构

有好几种方法都可以用来寻找以单边为梳齿的梳子结构。除了上面介绍的现成捷径之外，也有一些更精巧的启发式算法，还有一种具有多项式运行时间的恰当分离方法。这些方法都很成功。研究者也发扬正宗的算法工程风格，充分利用试探算法寻找一般性的梳子。他们的核心思想是，先把城市构成的簇换成单个顶点，得到收缩图，然后再在其中搜寻以单边为梳齿的梳子。找到梳子以后，可以再反向展开，扩充为原图中一般性的梳子结构。这里的技巧在于簇的创造性选择，通常要借助一定方法，在线性规划解中搜索边界取值接近2的集合。在计算42座城市的TSP题目时，最后给出那把有3个梳齿的梳子的求解方法正是属于这一类启发式收缩聚类算法。

6.3.3 不自交的线性规划解

表面上，探索梳子分离算法的任务是没人想干的脏活累活，但实际上，这方面的研究会从其他各研究领域借鉴方法和结构，因此有可能发展成为非常有趣的数学研究。Lisa Fleischer和Éva Tardos在康奈尔大学的研究工作就是出色的例证，研究时间是20世纪90年代末期。[1]对算法工程的紧迫需求涌现自计算方面的研究，主导了她们所处学术领域的研究方向。她们没有随大流研究试探风格的启发式算法，而是经过冷静思考，得出了优美精确的理论结果。这也是针对一般性梳子结构的第一个准确结果。

1. Fleischer, L., é. Tardos. 1999. Math. Oper. Res. 24, 130–148.

Fleischer-Tardos研究的主旨是把重点放在满足特定条件的线性规划解上：这类解可以画成各边互不相交的图，例如之前看到的环游美国42座城市路线以及图6-15的1000座城市周游路线。这样的限制条件很有用。几何形式的TSP题目多数都有完全或近乎不自交的线性规划解。图论里有些工具虽然不能用在一般情形下，却适用于此类情况。她们通过分析，给出了一种分离算法，其运行时间为多项式时间。不过这种算法得到的梳子究竟质量如何，我们不能妄下结论。

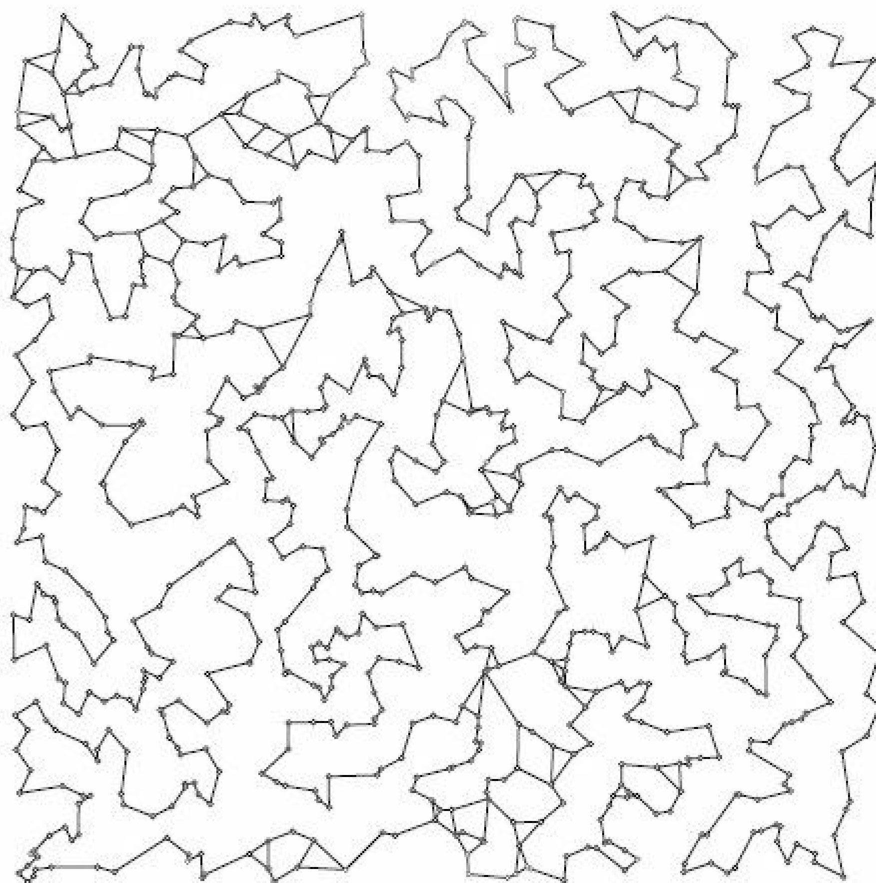


图**6-15**　子回路线性规划的松弛解，题目包含**1000**座城市

已经知道，每条回路与*k*齿梳子的边界至少相交3k+1次。另外，如果线性规划解满足所有子回路消去约束条件，相应的相交次数取值就至少为3k次，也就是说梳子不等式即便得不到满足，最多也只会相差1。现在给定一个不自交的线性规划解，只要存在一个梳子结构使二者相交次数取值为3k，就可以保证Fleischer-Tardos算法必定能给出一个这样的梳子结构。然而，如果梳子不等式的违背程度不到1，而是1-δ，其中δ是一个很小的数，那么该算法未必能保证找到这样的梳子。因此，该算法令人喜忧参半。一方面，它能找到违背不等式程度最大的梳子，这是件好事；可是另一方面，它找不到梳子时，有可能依然存在许多梳子结构使不等式不成立，从务实的角度来看，漏掉这些梳子当然是件坏事。

能够完全解决不自交线性规划解的梳子分离问题的多项式时间算法，应该如何构造？这一研究问题依然有待解决。它的答案不但将成为重大的理论成就，而且很可能直接影响TSP的实际计算。英国兰卡斯特大学的Adam Letchford对此作了探究。他采用了Fleischer-Tardos研究的

思想，只不过改变了思路，自下而上建立起一类新的约束条件，得到了不自交情形下的多项式时间分离算法。[2]Letchford的一系列约束条件包括梳子结构和许多其他结构，统称为多米诺奇偶性不等式（domino-parity inequality）。

2. Letchford, A. L. 2000. Math. Oper. Res. 25, 443–454.



图**6-16　Lisa Fleischer**（左）、**Éva Tardos**（中）和**Adam Letchford**（右）

“许多其他结构”听起来好像不错，但是它的实际后果并不清晰。根据已有的海量计算研究，我们知道，梳子不等式的力量非常强大，但是Letchford的多项式时间算法给出的约束条件更为一般，有时候虽然是当前解不满足的梳子不等式，但并不是小平面定义不等式。几年后，Sylvia Boyd、Sally Cockburn和Danielle Vella组成的加拿大研究小组解决了这个问题。他们把人工计算和计算机程序实现结合起来，证明Letchford算法用在中等规模的测试题目上非常高效，结果令人印象深刻。[3]在Daniel Espinoza和Marcos Goycoolea的领导下，佐治亚理工学院的研究小组紧随其后，完成了一项大型计算研究，全面实现了该算法的自动化。[4]这项研究成果成为了Concorde程序的重要新增模块，大大提高了TSP求解速度，最终解决了85 900座城市的题目，创下TSP实战领域的纪录。

3. Boyd, S., S. Cockburn, D. Vella. 2007. Math. Program. 110, 501–519.

4. Cook,W., D. G. Espinoza, M. Goycoolea. 2007. INFORMS J. Comp. 19, 356–365.

6.4 Edmonds的“天堂之光”

诚然，不等式质量越高，给出的路线长度下界就越准确，相应的计算机程序运行速度也就越快。可是，从不等式切入的这条研究路线最终能否通向百万美元大奖的领奖台呢？我们有一个先例可供参考，那就是Jack Edmonds对完美匹配问题给出的惊天妙解[1]。

1. Edmonds, J. 1965. Canadian J. Math. 17, 449–467.

图的完美匹配就是将顶点两两配对的一组边。换言之，每个顶点都恰好是完美匹配中的唯一一条边的端点。如果把代价值赋给图的各边，则完美匹配问题就是寻找完美匹配，使其中的各边代价之和最小。这道题和TSP一样，很难发现高效解法。

既然我们不知道怎样才能求解这道最优化问题，最好请线性规划对偶性出场。事实上，完美匹配满足一种形式的度约束条件，具体说来，每个顶点的度（与顶点相连的边的总数）都应该是1。这个切入点不错，我们还需要合适的割平面。Edmonds提供了所需的割平面。他提出的带花树开花不等式（blossom inequality）表明，顶点数目为奇数的图不可能存在完美匹配。因此，图中任取一个顶点数目为奇数的簇，完美匹配中必须至少有一条边连接该顶点簇与图的其余部分。相应的线性不等式与子回路消去不等式左端形式相同，但前者只要求取值至少为1，而后者要求至少为2。

Edmonds一次性把所有开花不等式都添加到线性规划模型之中，每个奇数集合S都对应一个不等式，也就是一个割平面。然后他证明了，得到的多面体确实是图的完美匹配的凸包。利用这种描述，他得以直接使用线性规划对偶性，构造出一个多项式时间算法，无需逐步选择割平面就能计算得到最小代价完美匹配。这个结果非同寻常，他自己总结道：“这有一个好算法，这有一个已经攻克了的整数规划问题。要知道，这是一场布道，一场货真价实的布道。这有一个已经攻克了的整数规划问题。这是我第一次看到天堂之光。”[2]

2. Edmonds (1991).

他的结果能不能用来解决TSP呢？1964年，Edmonds探讨得出，TSP多面体顶点虽然数目庞大，但是有统一的简单描述，因此小平面可能也有简单描述。他写道：“至少应该对此抱有希望，因为找到相当出色的旅行商问题算法无疑等价于找到这样的简单描述。”[3]他的断言的确大胆，却准确洞见了大奖的方向。事实上，1979年，Khachiyan的线性规

划算法登上了《纽约时报》的封面，而它恰恰具有一个有趣的性质：无需列出线性规划问题的显式约束条件，只要有可用的分离程序，就可以执行这个算法。[4]因此，Khachiyan算法只需服从几个技术条件，就可以用来证明：根据好的分离算法可以得到好的最优化算法；反之亦然，根据好的最优化算法也可以得到好的分离算法。这个数学结论非常深刻，最完整的形式于20世纪80年代获得证明，证明者是Martin Grötschel、László Lovász和Alexander Schrijver。[5]

3. Gomory (1966).

4. Khachiyan算法基于通用算法“椭球法”（ellipsoid method），后者由David Yudin和Arkadi Nemirovski发明。

5. Grötschel, M., L. Lovász, A. Schrijver. 1993. *Geometric Algorithms and Combinatorial Optimization*, 2nd edition. Springer, Berlin, Germany.

由此推知，要想解决TSP，必须给出路线凸包的描述，还要配备多项式时间的分离算法。漂亮的推论！从务实的算法工程角度，TSP需要用到更快、更好的分离算法，而这种方法同样也与克雷数学研究所的百万美元大奖紧密联系在一起。如果能够实现就太棒了！我们将再次目睹天堂之光！

6.5 整数规划的割平面

天堂谈得差不多了，下面脚踏实地，考虑一下整数规划问题。成功的整数规划方法都扎根于TSP研究，割平面法也不例外。它绝对是新型整数规划解题程序里最重要的工具。无论是子回路约束条件，还是梳子不等式，抑或团树不等式，都是专门用来解决TSP的；但是，通过割平面逐步改进线性规划松弛解的整体方法却是通用的，它同样可以用来解决一般的整数规划问题。

Ralph Gomory是第一个认真研究整数规划割平面的人。后来，他几度升职，在IBM公司担任高级副总裁，负责科技部门，又在美国斯隆基金会（Alfred P. Sloan Foundation）担任主席。20世纪50年代中期，Gomory还在普林斯顿大学数学系，只是一名默默无闻的博士后学生。当时，他在研究如何从线性规划问题中分离出整数值的解。他学习过古典数学，因此求解时力图使用丢番图分析理论，即不定方程理论。丢番图分析（Diophantine analysis）就是对线性方程（组）整数解的研究。对线性规划模型里用到的线性不等式进行推广似乎会有效果，但是没日没夜地奋战了整整一周之后，所有的结果依然不过是一堆只做出来一半的题目。[1]

1. Gomory, R. E. 2010. In: Jünger et al., eds. *50 Years of Integer Programming 1958–2008*. Springer, Berlin. 387–430.

> 第八天傍晚，我已经无计可施了。可是我依然相信，对于一切给定具体系数取值的不定方程题目，如果有必要给出整数解，那么我肯定总有办法给出来的。当时，我暗暗问自己：假如你真的必须解决某道具体的题目，必须想方设法求出答案，那么你第一步会做什么？我的第一反应就是，第一步会解决相应的线性规划（最大值）题目，如果答案解出来是7.14，那么我至少知道整数解最大不可能超过7。我心里刚想到这个显然的结论，马上感觉左脚两只脚趾突然发麻，当下我就意识到，这是个非同寻常的结论，在经典丢番图分析里绝对没出现过。

上面的思想是说，如果知道一道线性规划题目的所有解都满足不等式$3x+2y\leq 7.14$，那么我们就知道所有整数解也都满足$3x+2y\leq 7$。因此，所有与线性规划可行域相切的半平面都可以再推进一点点，而不会“切割掉”任何整数解。[2]

2. 20世纪70年代初期，对右侧向下取整以加强线性不等式的思想由Vašek Chvátal发展为精

细的理论。

Gomory根据这个结论，研究出了一种解决纯整数规划问题的算法。所谓纯整数规划问题是指所有变量都必须取整数值的线性规划问题。Gomory割平面法借鉴了单纯形算法的形式，每当字典给一个变量赋非整数值时，就推导出一个割平面。他的论文篇幅虽然短小，却在几年内彻底颠覆了整数规划领域的研究局面。可惜，后来计算机速度提高，能够解决大型题目了，Gomory割平面法在实际应用上表现不佳的劣势也变得一览无遗。即便如此，这段故事的结局还是很美好：Gomory割平面法中，构造割平面的机制同样出自他之手，而如今的整数规划解题商业软件用到的正是这种机制的变体。

第7章 分支

> 基本方法就是把所有路线构成的集合拆分成越来越小的子集，分别计算每个子集的最优路线代价下界。
>
> ——John Little等人，1963年[1]

1.Little, J. D. C., et al. 1963. Operations Research 11, 972–989.

在1954年Dantzig等人的研究工作之后，直到1973年Chvátal的成果发表之前，是TSP割平面法的蒙昧时代。当时，研究人员倾力研究各种各样的其他解法。其中，最重要的解法是分支定界法（branch-and-bound）。该方法以分而治之为理念，和割平面法一样，都是在旅行商问题的背景下提出的通用方法。如今，最先进的TSP计算软件结合了分支定界算法和割平面法，从而有能力解决包含成千上万座城市的旅行商问题。

7.1 拆分

在线性规划的松弛问题里寻找最优路线，就像是在干草堆里寻找最精细的针一样。如果有充足的不等式和耐心，割平面法也能解决问题，把上面的干草都移走，让那根要找的针最终脱颖而出，在干草堆顶上闪闪发光。

虽然听起来不错，可惜有时候事情并不尽如人意，有可能每次新加入割平面都只能移走一点点干草。此时，我们便可以放弃继续在剩下的干草堆上“割”来“割”去，转而考虑把大草堆“分”成两垛小草堆。如果拆分方式合理，那么解决新得到的两个子问题就会比原先搜寻整垛大草堆更容易，也就能在干草堆和所有针之间理清头绪。这个拆分步骤称为分支（branching）。Dantzig等人描述了一般情况下的分支思想，后来Willard Eastman根据该理念提出了完整的TSP算法[1]。

1. Eastman, W. L. 1958. *Linear Programming with Pattern Constraints*. Ph.D. thesis. Department of Economics, Harvard University, Cambridge, Massachusetts.

我们将在下一节介绍分支定界算法。这里首先以42座城市的环游美国问题为例，深入探讨其中的一步分支操作。对于这道题目，本来只用割平面法就足以轻松给出答案，所以，我们人为施加限制，只允许使用一类割平面，也就是对应于线性规划解中孤立子图的子回路不等式。这样一来，第6章列出的前三步计算仍然可以照常进行，得到的下界为682.5个单位长度，连通图如图7-1所示。

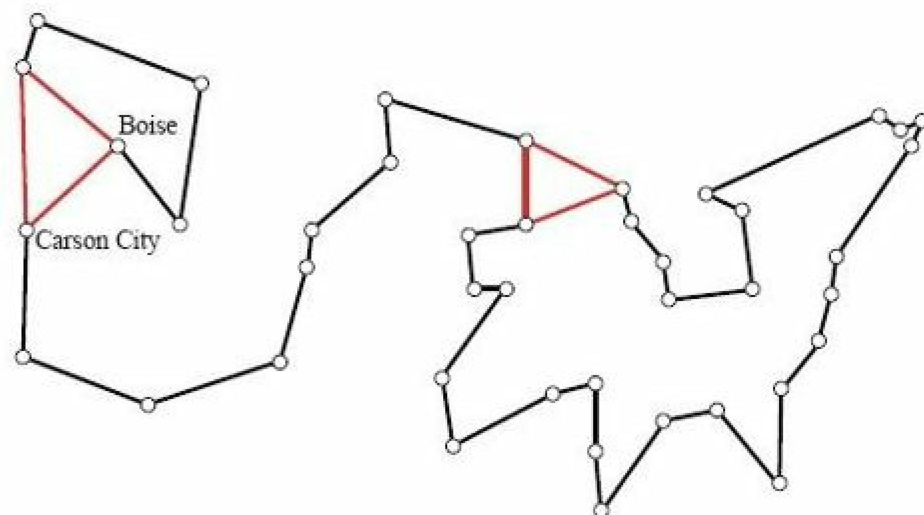


图**7-1**　增加**3**个子回路消去约束之后求得的解

至此，图7-1中的线性规划解相当于草堆顶部露出来的一束干草。因

此，合情合理的分解问题方式至少应该把这个解移走。标准做法非常简单，只需要选取一条红边，并要求题目必须在“使用”与“不使用”该红边之间做出选择。也就是说，我们把路线集合一分为二，用到该红边的路线属于第一个子集，而不包含该红边的路线则属于第二个子集。在线性规划模型里，这种方法效果很好，因为只需要加入约束条件x_{ij}=0就能得到第一个子问题，加入x_{ij}=1则能得到第二个子问题，其中城市i和城市j是该红边的端点。新生成的两个子问题分别称为分支的0侧（0-side）和1侧（1-side）。

在环游美国之旅的例子中，我们选择Boise和Carson City之间的边作为分支边，得到的线性规划解如图7-2所示。请注意用黄色区域标出的这条分支边，分支0侧的线性规划解里不含该边，而分支1侧的线性规划解里含有该边。单纯形算法再一次给出了正确答案。0侧解对应的目标函数取值为687.5，而1侧解对应的取值为676。由于每条周游42座城市的回路同时都是两个线性规划模型的可行解，我们知道，所有路线取值都不可能低于686个单位长度。

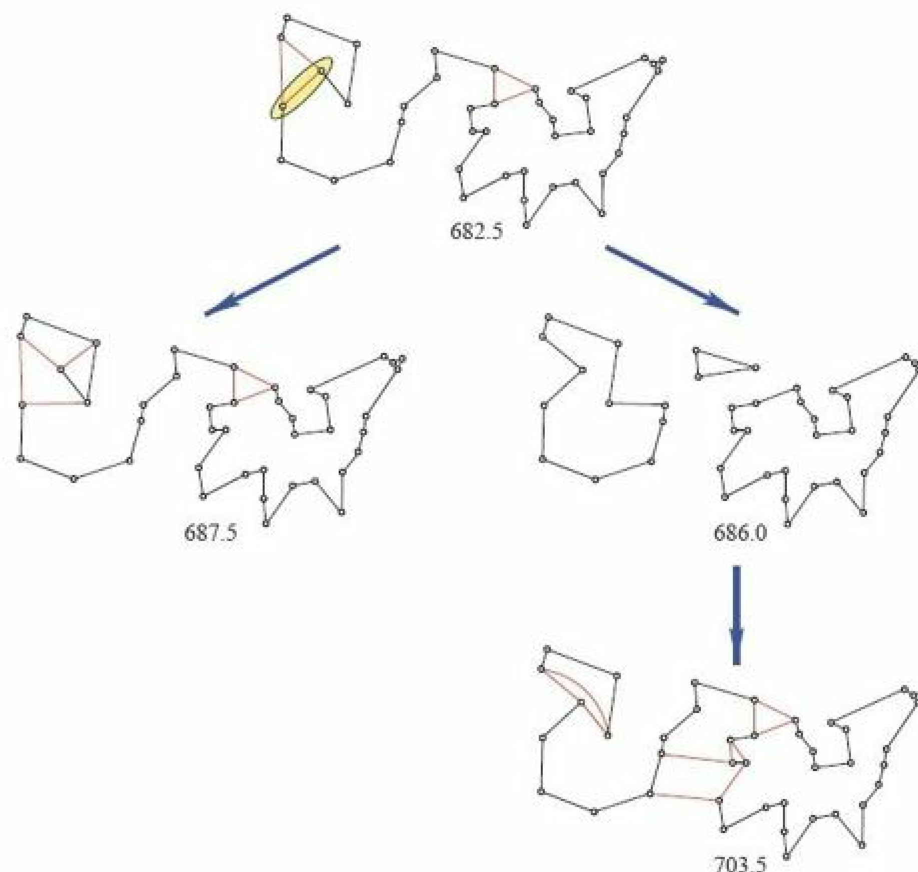


图**7-2**　选择连接**Boise**和**Carson City**的边进行分支操作

这一步分支操作不但把解的下界从682.5提高到了686，而且还有进一步的好处：在1侧的线性规划解中有几个孤立子图，只需加入几个子回路消去约束就能收拾干净，得到一个新的线性规划解，长度为703.5，见图7-2的右下角。很好。我们既然已经知道有条路线长度只有699，那么继续寻找潜在更优路线的过程中，就没有必要再看分支的这一侧了。由此，我们可以直接去掉1侧的子问题，也就是“修剪”掉一个分支，以后搜寻最优路线就不再考虑这一部分的解。

总而言之，我们首先把干草堆一分为二，接着向模型中添加若干个割平面，然后在两个小草堆中舍弃一个。并不是所有时候都能出现这么理想的形势变化，不过这个例子应该足以让你相信，在TSP求解过程中，分支法的确能够发挥很大的用处。

7.2 搜索队

Eastman的研究工作带来了一种搜索策略，TSP研究者Little等人将其命名为分支定界法。[1]思路非常直截了当。首先从原问题出发，原问题也称为根松弛问题（root relaxation）。对于某个子问题，只要由它得到的线性规划下界大于等于一条已知路线的长度，就舍弃该子问题，以后均不予考虑。每一步，我们都选择一个剩下的子问题，进行分支操作，得出两个新的子代的子问题。以上过程重复进行，直到每个未分支的子问题均被修剪舍弃为止。

1. Little, J. D. C., et al. 1963. Op. Res. 11, 972–989.

分支定界法说起来简单，但受到很大局限，只有优秀的定界机制才能保证计算成功。弱的下界会导致搜索树非常庞大，反之，强的下界则能实现快速修剪舍弃，从而快捷求得最优解。Eastman本人并没有考虑到要用割平面法改进度约束下界，因此他的计算结果虽然成功，规模却不大。在博士论文里，他只解决了一道10座城市的例题。

相比之下，图7-3的规模则大得多。该图展示了如何利用分支定界法解决环游美国42座城市的问题，根松弛问题是长度为687.5的线性规划模型，每次在未修剪的子问题中出现子回路时，都加入约束条件使之消除。我们效法Eastman的先例，用树的形式展示整个搜索过程，子代的子问题与父代的子问题之间直接相连。请注意，每步分支操作之后，子代子问题对应的线性规划下界都会产生实质性改进。这样的结果正符合我们的心意。

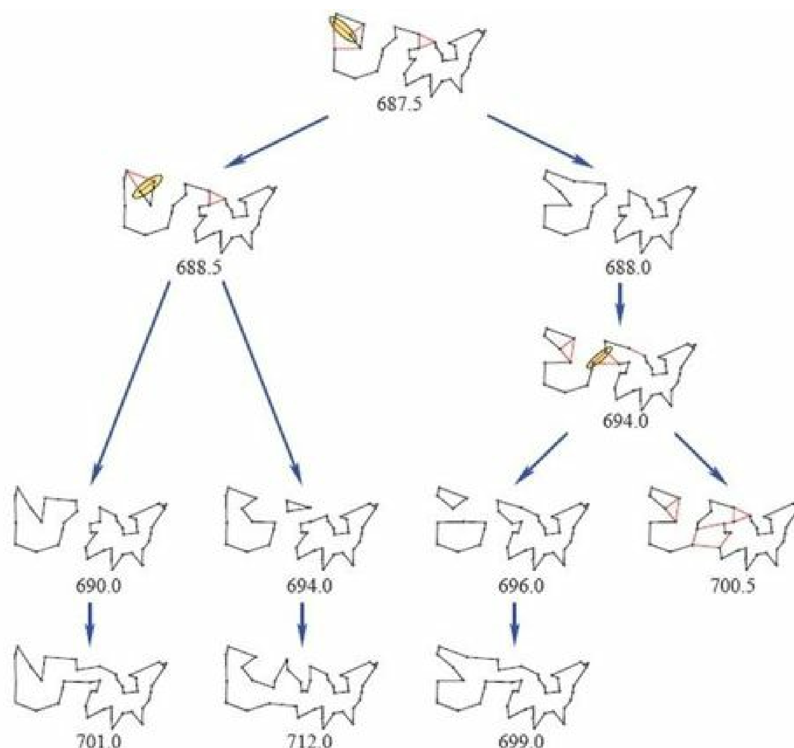


图**7-3** 环游美国**42**座城市问题的分支定界搜索树

7.2.1 分支切割法

尽管割平面法和分支定界法起初曾经彼此较劲，这两种方法本质上却是天生一对。实际上，20世纪80年代，研究者将两者结合起来，一时轻松横扫天下。当时的领军人物是Manfred Padberg和Giovanni Rinaldi。他们创造了“分支切割法”（branch-and-cut）这个新词，给出了谨慎细致的程序实现，让旅行商的计算过程突飞猛进。随着一道2392座城市的测试实例宣告破解，他们突破了之前的所有计算纪录。[1]

1. Padberg, M., G. Rinaldi. 1987. Oper. Res. Let. 6, 1–7.

Padberg和Rinaldi在计算中并没有对割平面法施加人为限制，而是恰恰相反，尽情使用所有能用的割平面，想方设法改进线性规划模型。他们由此发现了一种有用的处理方法，就是建立一个库（pool），把每次切割都保存在其中，以供所有子问题共用。换言之，如果一个割平面能够改进一个子问题的线性规划下界，那么研究者就把它存起来，等到处理其他子问题时也许能派上用场。建立这个库是出于两方面的考虑。第一，搜索该库的速度会比执行复杂分离算法的速度快得多。第二，TSP计算用到的绝大多数分离算法都是试探性的启发式算法，有时它们试探

不中某道子问题，却能试探中另一道子问题。因此，我们可以通过把不等式集结成库，整理出所有正中目标的试探，看看它们能对搜索树的其他部分起到什么作用。

7.2.2 强分支

Vašek Chvátal喜欢把分支步骤比作结婚。一旦决定进行分支，就意味着承诺使用分支后的新观点看待问题。分支之后，便不可能回头。按照他提出的比喻方式，可以认为，分支定界法的早期研究者一般都使用一见钟情式系统，会与路上遇到的第一个潜在结婚对象结为伴侣。这也就是说，他们只要看到线性规划解中有条边取分数值，就会立刻以这条边为分支标准，将原问题一分为二，得到一对子问题。这种分支方法速度够快，但是在分支定界法里，定界过程也要耗费大量时间，所以有必要多花点工夫，认真选择分支边。Padberg和Rinaldi的做法用到了一种技术，可以比作利用婚姻介绍所甄选潜在的伴侣。在Concorde程序里，我们又加了一步，计划在做出选择之前，首先跟数名候选对象出去约会几次再说；然后我们“全面发展”，决定在最终正式进入婚姻殿堂（分支）之前，先和最好的几个人选同居一阵子再做结婚承诺。

下面从第一步的婚姻介绍所开始解释。Padberg和Rinaldi采用统计学方案，构造出一组分数化程度最高的边，即取值距离0和1都很远的边。在这一组边中，他们选出旅行成本最高的一条边。此举理由充分，因为与短边相比，在长边上进行分支往往对线性规划的界影响更大。

Concorde程序的思路称为强分支（strong branching）：对于单纯形算法可能得到的数值，要求线性规划解题程序给出强有力的提示，从而降低选择步骤的猜测程度。对于候选分支边对应的每一对子问题，解题程序执行有限次数的单纯形换主元操作，比如进行50步，据此给出提示。我们参考并比较这些数值，判断哪种情况对线性规划的界改进最为明显，就选择相应的分支。这步计算过程可能花费相当长的时间，但只要能避免错误拆分问题导致搜索树大小加倍，即便多花些工夫也是值得的。

“全面发展”之所以能改进结果，是因为这一步选定了强分支挑出的少数优秀人选，然后实际解决相应的线性规划子问题，包括对于每个可能的子问题应用有限个割平面。该过程能准确预期得到的线性规划的界，但是需要付出高昂的计算代价，只有最难的例子才值得如此求解。正统数学家大概会说，这样做是在投机取巧，因为最终做决定之前已经

试验过各个分支的结果。可是为了对抗旅行商，我们所做的一切都属于光明的正路。

7.3 整数规划的分支定界法

分支定界算法植根于TSP研究，诞生后不久就成功打入了一般的整数规划领域。整数规划分支定界法的研究先驱是Ailsa Land和Alison Doig，她们来自伦敦政治经济学院。[1]

1. Land, A. H., A. G. Doig. 1960. Econometrica 28, 497–520.



图**7-4　Ailsa Land**，英国班夫郡，摄于**1977**年

在2010年出版的回忆录里，她们两人讨论了当年提出的算法程序。[2]

2. Land, A. H., A. G. Doig. 2010. In: Jünger et al., eds. *50 Years of Integer Programming 1958–2008*. Springer, Berlin. 387–430.

> 起初，我们并没有把该方法看成是“分支定界法”，而是考虑了它的“几何学”解释，即对线性规划约束条件定义的凸包可行域进行探索。我们不太清楚，此前文献中是否已经提到过“分支定界法”，不过就算前人提到过，我们也并没有想到使用这个名称。还记得Steven Vadja说他见过一些通过“Lawndwa”解决整数线性规划问题的法国人。我们意识到所谓“Lawndwa”是“Land-Doig”的法语读音，由此认为法国人也并不知道“分支定界”这种说法。

Land-Doig的基本方法主导了整数规划计算的实际应用，在一定意义上，这是基于割平面的Gomory算法未曾取得的成就。直到20世纪90年代中期以后，分支切割法才成为整数规划商业软件的主力，标志着两大竞争对手终于开始携手合作。

第8章 大计算

> 我认为，自己的所有理论成果带来的兴奋感，都比不上Held和我首次测试定界方法的那一夜，我们看到电脑上无数数字奔涌而出时的激动心情。
>
> ——Richard Karp，1985年[1]

1. 1985年图灵奖获奖演讲，见Karp (1986)。

数学方法不断进步，算法工程精益求精，计算平台日益强大，三者结合起来，引领TSP走向前所未有的显赫高位。然而，战斗尚未成功，前路依然漫漫。让我们回顾与旅行商斗争的历史，看看今天走到了哪一步。

8.1 世界纪录

说到TSP世界纪录，毫无疑问必须把Dantzig、Fulkerson和Johnson的研究列在首位。

> 对这么大规模的TSP题目，三位作者都能够通过手工计算找到最优解并证明其最优性，实在是太惊人了。
>
> ——George Nemhauser和Martin Grötschel，2008年[1]

1. Grötschel, M., G. L. Nemhauser. 2008. Discrete Optim. 5, 168-173.

> Dantzig、Fulkerson和Johnson展示了求解大型TSP题目的一种方法，后面的解法都只是锦上添花而已。
>
> ——David Applegate等人，1995年[2]

2. Applegate, D., et al. 1995. DIMACS Technical Report 95-05. DIMACS, Rutgers University.

后来，世人终于领会了兰德小组的研究工作，把他们的技术推广到不同的应用方向，得到了其他重要成果。但是若论创新思想之丰富，影响作用之广泛，系统阐述之水平，他们发表于1954年的原始论文始终是旅行商问题研究史上最伟大的成就。

8.1.1 随机选取的64个地点

紧接Dantzig等人的工作之后，TSP研究前线风平浪静了好几年。虽然人们尝试了五花八门的方法，可是计算测试的题目一般仅限于10座城市上下的规模。到了1971年，Michael Held和Richard Karp终于迈出一大步，开始推进规模更大、难度更高的计算，给这段枯竭的岁月画上了句号。[1]

1. Held, M., R. M. Karp. 1971. Math. Program. 1, 6-25.

在1985年的图灵奖获奖演说（Turing Award Lecture）中，Karp明确陈述了Held-Karp研究的用意。“数年前，兰德公司的George Dantzig、Raymond Fulkerson和Selmer Johnson，结合人工计算和自动计算，曾成功解决一道49座城市的问题。而我们希望打破他们的纪录。”[2]

2. Karp (1986).

他们确实打破了纪录。Held和Karp先是再次解决了同一道49座城市的题目，接着继续解决了一道环游美国57座城市的题目，又攻克了一道

64座随机城市（采用直线距离）的题目。获胜法宝是基于复杂定界机制的算法和程序。定界机制源自生成树，得到的界则整合到分支定界搜索过程中。该方法用在测试题目上，能产生非常小的搜索树。[3]

3. Interview with Richard Karp, October 24, 2009.

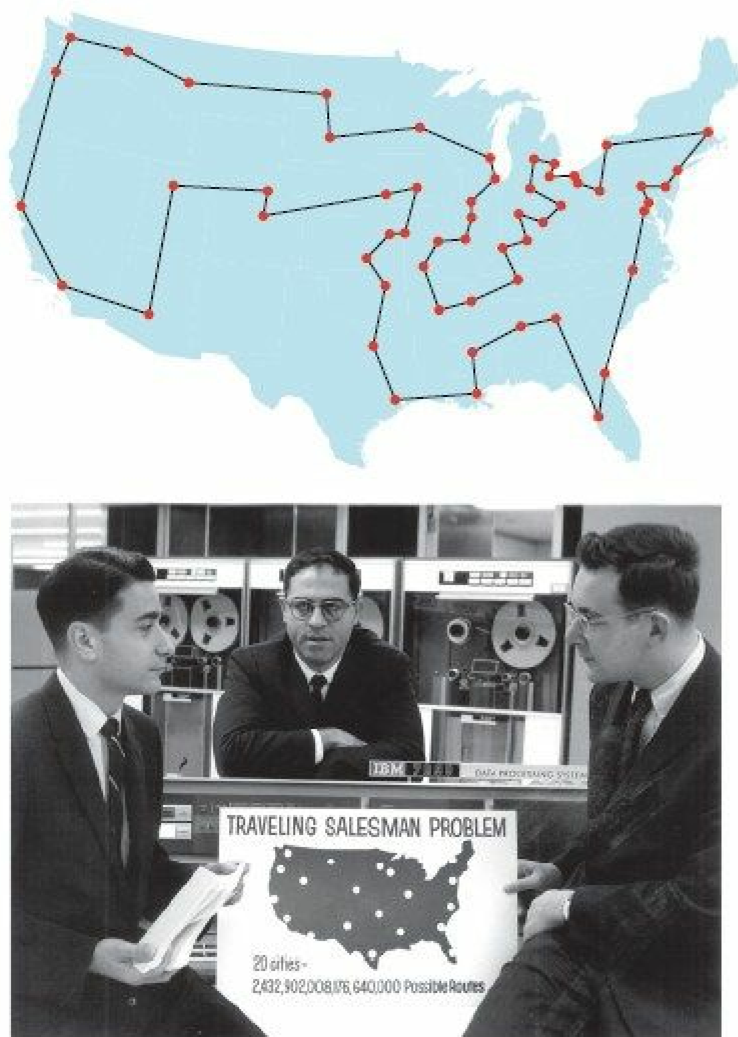


图**8-1**　上图：包含**57**座城市的**Held-Karp**最优路线；下图：**Michael Held**、**Richard Shareshian**和**Richard Karp**，**1964**年。**IBM Corporate Archives**提供图片及照片

> 真是振奋人心的结果。此前我们历尽艰难险阻，如今突然如有神助。分支定界过程中，生成的树往往相当狭小，偏差很少，给出的界限也非常好，几乎直接就能得到最终解。

在所有创纪录的TSP计算中，Held-Karp研究是独一无二的，因为它

并没有直接利用割平面法。不过，他们用到的定界机制确实与线性规划及Dantzig等人的成果有联系。事实上，子回路线性规划的松弛问题中，目标函数的最优值可由Held-Karp法的界给出近似。而且，该方法的全过程可以视为绕开一般线性规划解题软件的做法。他们避免使用线性规划解题程序，并不能反映出当时此类程序质量如何，只是运行割平面法必须要迭代求解，而当年可用的软件确实很难如此使用。

8.1.2 随机选取的**80**个地点

后来，几个研究小组对Held-Karp的分支定界法作了局部细节修正。1975年，意大利的一个研究团队成功借此解出了一道采用直线距离的题目，规模提高到了67座城市。[1]但是，割平面法注定要卷土重来，毕竟除了子回路消去约束以外还有其他不等式，它们的力量非常强大，可以大大提升Held-Karp法得到的界。

1. Camerini, P. M., et al. 1975. Math. Program. Study 3, 26-34.

在Ailsa Land的领导下，伦敦政治经济学院的研究小组实现了反击。她的学生Panagiotis Miliotis结合割平面法和一般的整数规划，解决了一系列随机选取地点的题目，测试题目采用直线距离，最多可达80座城市。这种综合解题方法最早是由Glenn Martin在20世纪60年代中期提出的。[2]求解过程大致如下。首先，从度约束线性规划的松弛出发，限定所有变量取值只能为0或1，应用Gomory的整数规划割平面算法。如果得到的解是条完整回路，则它必定是最优解；否则，找出已知解违背的若干个子回路消去约束条件，加入模型中，重新调用Gomory算法，重复该过程。

2. Miliotis, P. 1978. Math. Program. 15, 177-188.



图**8-2　Panagiotis Miliotis**，**1974**年

Miliotis求解80座城市的题目时，所需的总运行时间只有不到一分钟。这意味着即使题目规模更大，也能够解决。然而，Miliotis却说出了下面的评语。

> 不幸的是，利用割平面的过程需要占用A矩阵（约束条件系数矩阵）大量空间。一道随机选取90座城市并采用直线距离的题目没能获得解决，这是正常的，因为A矩阵中非零元素的数目超过了30 000，而这些元素大部分都出现在割平面约束条件里。

Gomory法的这一特性实在令人扼腕，每加入一个割平面，约束条件里都包括线性规划模型的几乎所有变量，最终会严重拖慢单纯形算法的速度，以至于无法取得进展。相比之下，Dantzig等人的方法简单而又美好，其中的TSP不等式往往会把多数边赋值为0，产生的线性规划模型依然比较容易解出答案。

8.1.3 德国的**120**座城市

下一个TSP纪录的创造者是Martin Grötschel。他的方法是纯粹的割平面法：他求解模型使用了线性规划解题软件，但寻找不等式只用了手工计算，颇得1954年兰德小组研究的真传。我们已经在第1章见过他的成果，图1-9的三条周游德国的路线里，经过120座城市的那一条就是他的最优路线。

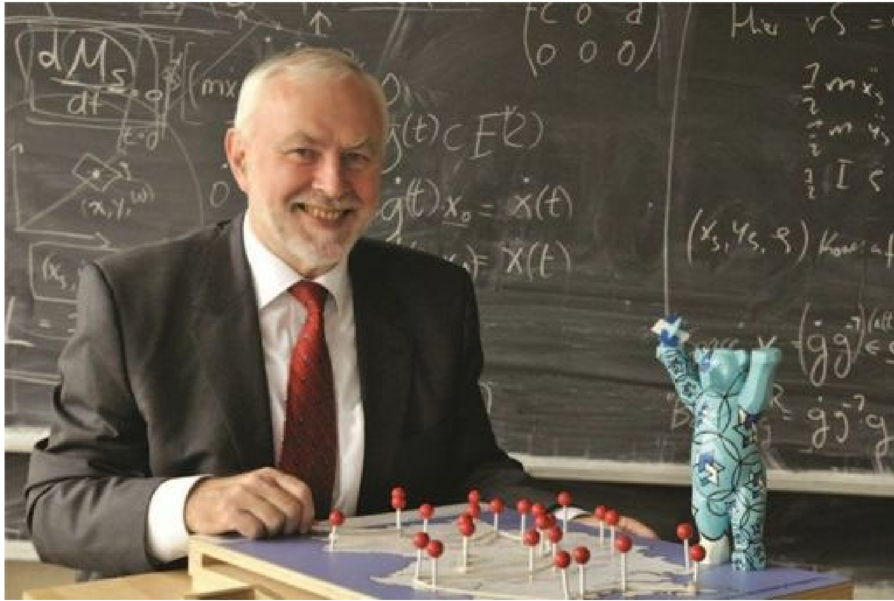


图**8-3　Martin Grötschel**，**2008**年，照片由德国柏林祖斯研究院提供

在计算这条破纪录的路线的过程中，Grötschel解决了13道线性规划的松弛，总共用到了36个子回路消去约束和60个梳子不等式。每道线性规划的松弛都给出一个界，手工计算用时在30分钟到3小时之间，而换成计算机时间则相当于30秒钟到2分钟之间。[1]他使用的线性规划解题工具是IBM的MPSX套装软件。在2005年6月11日的一封电子邮件里，他如下描述了计算的全过程：

1. Grötschel, M. 1980. Math. Program. Study 12, 61-77.

MPSX运行一次之后，我把解打印出来，画出对应的路线图。然后我复制几分副本，想办法寻找割平面。当然，积累一定经验以后，我能找到路线违背的好多不等式。但是，当年MPSX无法解决规模很大的线性规划问题，所以我只能精心选择看上去效果不错的割平面，把数目限制在每次运行只加入5到20个的样子。

这是一项非凡的成就，展示出了梳子不等式用来求解大型TSP题目的强大威力。

8.1.4 电路板上的318个孔洞

Grötschel计算完120座城市的题目之后，没过多久，Manfred Padberg立即启动了一项联合研究。他的合作者是Saman Hong，刚刚从美国约翰霍普金斯大学毕业的博士生。他们的项目是将割平面算法自动化，取得

了计算方面的成功，不但能解决多达75座城市的题目，而且在其他题目上可以计算出很好的下界。[1]该研究中，规模最大的例题是一道318座城市的钻孔题目，此前由Shen Lin和Brain Kernighan考查过。

1. Padberg, M. W., S. Hong. 1980. Math. Program. Study 12, 78-107.

Padberg不满足于取得好的近似结果，而是在几年后继续探索上述Lin-Kernighan例题的解。这一次，他的合作者是IBM公司的Harlan Crowder。[2]

2. Padberg, M. 2007. Ann. Oper. Res. 14, 147-156.

> 有天晚上，我们把一切都准备好，让计算机运行一次，求解那道318座城市的对称TSP。我们估计得花好几个小时，于是就去了“侧门”餐馆吃饭，它距离IBM研究中心不远。吃完饭回来的路上，我们讨论了万一程序失败，可以新加入哪些“花里胡哨”的玩意儿。到了IBM研究中心，我们去计算机房看有没有完成输出，结果发现有。程序明确表示，它在不到6分钟的计算机时间内，找到了最优解！

这道318座城市的题目以及许多规模较小的题目之解，为Crowder-Padberg研究画上了圆满的句号。[3]

3. Crowder, H., M. W. Padberg. 1980. Management Science. 26, 495-509.

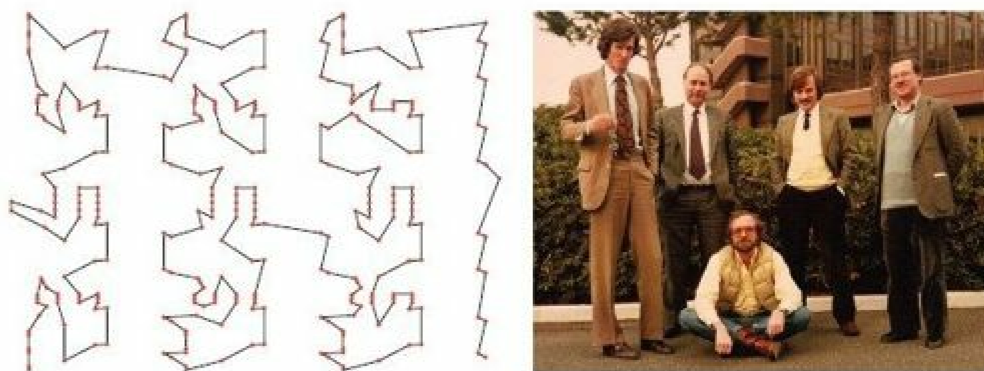


图**8-4**　左图：**318**座城市的钻孔问题的最优解；右图：**Manfred Padberg**（右二）和**Harlan Crowder**（前排坐者），**1982**年（照片由**Manfred Padberg**提供）

8.1.5 全世界的666个地点

下面，我们来到1987年。TSP计算研究再次掀起高潮，只有1954年的“大爆炸”可以与之相提并论。那一年公布了两项重大研究，第一项是Martin Grötschel和Olaf Holland攻克了大量测试题目，最难的一道包括从

世界各地选取的666座城市。看到数字666，《圣经》研究者会说它是“兽的数目”[1]。实际上，Grötschel选择这些城市时，确实有意创造出一道有如洪水猛兽般的TSP计算难题。[2]Grötschel和Holland结合使用了割平面法与一般整数规划，解决了这道难题。这一次，他们的整数规划求解程序是IBM的MPSX-MIP/370程序。他们之所以能获得计算上的成功，主要归功于新发现了许多启发式准确分离算法，可以用于梳子不等式，从而使程序得到非常强的线性规划松弛解。

1. 出自《圣经·启示录》3-18。——译者注

2. Grötschel, M., O. Holland. 1991. Math. Program. 51, 141-202.



图**8-5　Grötschel-Holland**周游**666**座城市的路线（左）；**Olaf Holland**（右），**2010**年

8.1.6 电路板上的2392个孔洞

无论是Glen Martin还是Panagiotis Miliotis，抑或是Grötschel和Holland，都综合使用了不止一种方法，用到了高效的整数规划解题程序。但是1987年的第二项重要研究却与众不同。Manfred Padberg和Giovanni Rinaldi的研究清晰地表明，完全把求解过程限制在TSP领域内也有诸多好处。他们使用的工具是分支定界法。他们的计算机程序解决了许多测试实例，包括周游美国532座城市的题目，前面提到的666座城市的Grötschel-Holland难题，还有分别包含1002座城市和2392座城市的钻孔问题。最后这道2392座城市的TSP题目宣告解决，意味着他们取得了一项惊天动地的成就，而且其计算显然也是截至当时最复杂的最优化问题解答过程。[1]

1. Padberg, M., G. Rinaldi. 1991. SIAM Review 33, 60-100.

Padberg-Rinaldi的研究引人注目，借助了大型的计算硬件，其中就有美国国家标准局的一台CDC Cyber 205超级计算机，还有IBM公司纽约

沃森研究中心的IBM 3090/600矢量计算机。但是改进TSP结果的动力其实是算法方面的工作，包括用于梳子和团树的新型启发式分离算法，以及分支切割法程序实现的众多新方法。他们在论文的最后给出了乐观的注记："在对称型旅行商的长篇传奇故事里，2392座城市的题目不会是最终结局。"

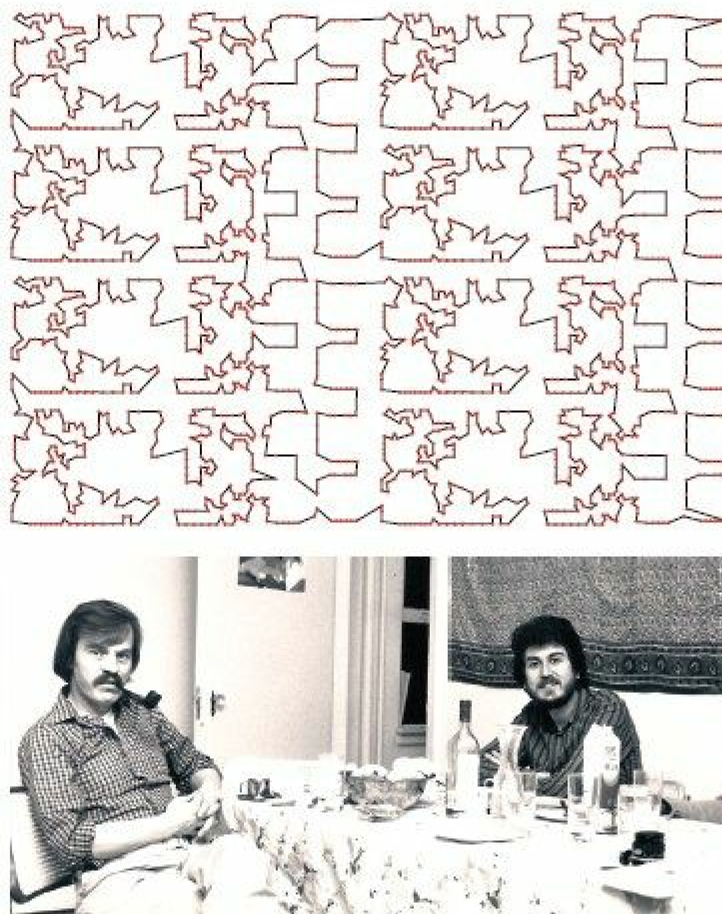


图**8-6**　上图：**2392**个孔洞的印制电路板的最优路线；下图：**Manfred Padberg**和**Giovanni Rinaldi**，**1985**年（**Manfred Padberg**供图）

8.1.7 电路板上的**3038**个孔洞

我和Dave Applegate、Bob Bixby、Vašek Chvátal在1988年开始合作研究，希望能续写传奇。最初，我们坚持不做别人做过的事情，试图避免使用割平面法。但是没过多久，我们就意识到自己犯了大错。到了1989

年，我们已经埋头于分支切割法的具体工作中。我们的第一版程序名为“Subtour”（子回路），因为编写它的初衷只是为了计算子回路线性规划的松弛之最优解，用来衡量不使用割平面法获得的界是好是差。后来，程序渐渐扩充增强，加入了一组新的分离例程，还用上了之前介绍过的强分支算法。

我们的计算研究基地是贝尔通信研究实验室（Bellcore），位于美国新泽西州。这里容纳有大约50台桌面工作站，当这些计算机的主人不在办公室时，我们就可以加以利用。每一台机器与Grötschel-Holland和Padberg-Rinaldi研究中用到的大型机相比，都完全不可同日而语，但是几十台构成网络联合起来，就有了惊人的速度。因此，把寻找割平面和处理子问题的任务分而治之，建立起一种并行处理方法，就是自然而然的事了。

不幸的是，觊觎空闲免费机时的并不只是我们这一群人。Arjen Lenstra是我们的对手，他领导了129位的数字RSA129的素因数分解工作[1]。争抢机时的过程不是彻头彻尾的友好竞争，不过最终双方团队都分到了为数不少的可用硬件。我们的策略简单明了：只在当前空闲的计算机上启动我们的软件，唯一例外是允许Lenstra同时运行他那烦人的分解因数程序，而且只要嗅探到机主回到工位上，就立刻终止程序。Dave Applegate编写了一小段程序，每秒检查几次是否存在用户输入，比如鼠标移动或者键盘敲击，从而实现了这种策略，而且让真正的机主极难发觉我们的手脚，他根本不知道我们接管了他的机器：如果他敲一条指令来看看机器在运行什么，那么指令的结果还没来得及发送到屏幕上，我们的程序就已经消失得无影无踪了。

1. RSA因数分解挑战（RSA Factoring Challenge）涉及的整数都是信息安全公司RSA（已被美国EMC公司收购）认为难以分解成素因数之积的数。RSA公司提供奖金，分别悬赏成功分解各数的人。

使用贝尔通信研究中心的上述网络，我们在1992年迎来了第一项成果，解决了一道3038座城市的电路板钻孔问题，题目出自Gerd Reinelt的TSPLIB题库。然后，我们在程序和算法中加入各种细微改进，在1993年得到一条周游旧时东德4461座城市的路线，又在1994年攻克一道7397座城市的计算机电路题目，先后刷新自己的结果。此时，我们达成一致，决定中止TSP项目，但是收尾过程并没有按照计划圆满完成。这或许是我们的幸事。

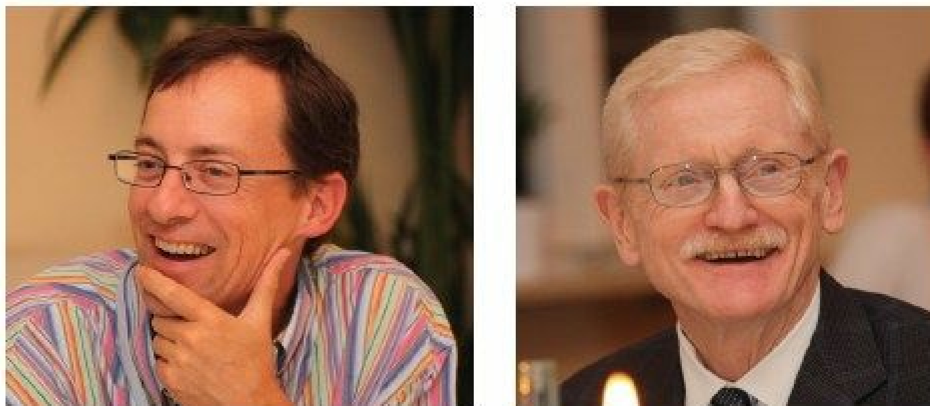


图**8-7**　左图：**David Applegate**；右图：**Robert Bixby**，由**Jakob Schelbert**拍摄，德国埃尔朗根，**2010**年

8.1.8 美国的**13 509**座城市

我们决定结束计算后，开始着手回顾多至7397座城市的TSP题目，整理记录其中用到的技术方法。这时，我们遇到了麻烦。既然我们自己都对研究细节不满意，想要说服其他研究者相信我们确立了可靠的解法，谈何容易！随着我们发现新的技巧，子回路Subtour程序也跟着时不时改进，但它不能体现出我们对于TSP计算的全局观点。有鉴于此，我们作出了唯一的理智选择——放弃。我们没有整理旧程序的文档，而是从零开始编写全新的程序，并称之为Concorde。

整个项目推倒重来非常难得，我们抓住这次机会，把针对规模大得多的TSP题目的算法和技术都整合进来。新增内容中，有一个“局部切割”分离例程非常关键，它依靠城市簇收缩来获得非常小的图，如此一来，某种基于线性规划的割平面搜索方法虽然耗时很长，也可以派上用场。

我们的计算研究主要意在解决一道13 509座城市的周游美国路线，并于1998年获得成功。此后的数年内，我们继续进行研究，但是主要目标则是理解消化已有的内容，就好比是赛车手逐渐熟悉新车一样。在此期间，我们于2001年计算出了周游德国15 112座城市的最优路线，并于2004年计算出了环游瑞典24 979座城市的最优路线。



图**8-8**　周游美国**13 509**座人口不少于**500**人的城市的路线

8.1.9 计算机芯片上的**85 900**个门电路

那道瑞典之旅题目的TSP计算，将Concorde的能力推向了极致。这项作业是在佐治亚理工学院完成的，用到了96台双处理器计算机构成的集群。程序作为后台进程，在机器没有其他活动时运行。总的计算机时间长达84.8年。

周游瑞典的路线颇令我们洋洋自得，但是TSPLIB里还有两道更大的题目，分别包含33 810座城市和85 900座城市，它们似乎超出了Concorde程序的能力范围。好在正当此时，Daniel Espinoza和Marcos Goycoolea对Letchford分离算法的程序实现出现在了网上（见6.3节最后的讨论部分），为我们带来了足够的马力，足以一试身手，挑战一下能否彻底解决Reinelt的TSPLIB题库里的所有题目。



图**8-9　Daniel Espinoza**（左）和**Marcos Goycoolea**（右）

2005年2月，我们开始对85 900座城市的TSP题目展开最后一轮程序运行。2006年4月，最优解诞生，宣告计算结束。线性规划的界稳步攀升，如图8-10所示，其中数据点取自几乎每天都有的计算日志，唯一的中断是2005年12月，当时恰逢计算机集群统一安排维修。终于有一天，计算出来的下界比当时已知的最好路线的长度只差不到0.001%了，那条路线是由Keld Helsgaun于2004年发现的。0.001%的差距已经足够接近了，使用分支切割法只需稍作搜索便能很快完成剩下的这一点工作，证明Helsgaun的路线就是真正的最优路线。

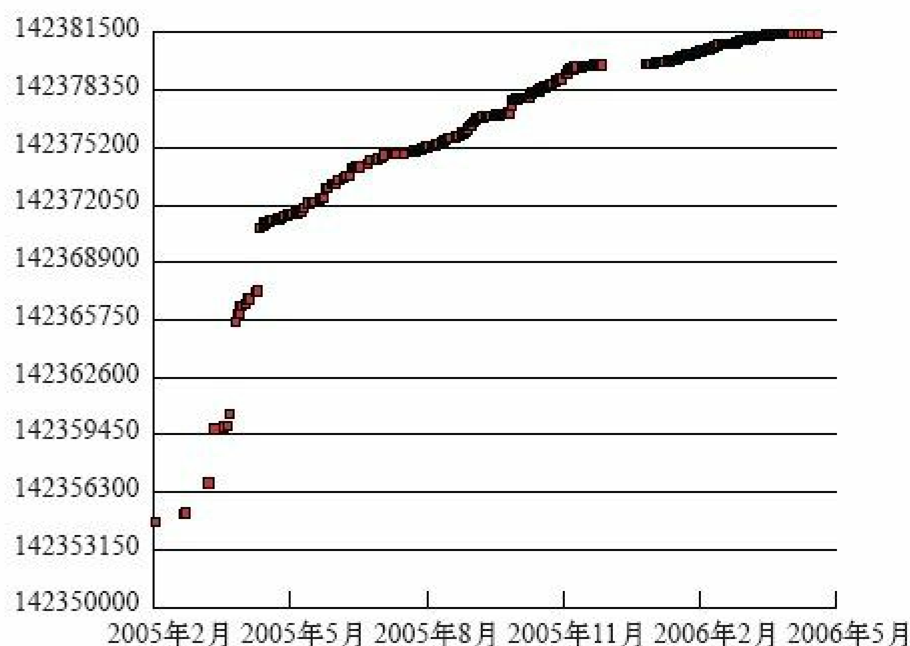


图**8-10　85 900**座城市的**TSP**题目计算中，线性规划的界不断改进

虽然我们自己声明已经解决了这道题目，但是计算耗费了136年的计

算机时间才完成，所以其他人当然都很难确切核实真相。因此，2009年，我们公布了一则计算机程序和数据集，保证了85 900座城市的路线的最优性[1]，与它的创纪录地位相一致。分支切割搜索步骤中，每一道子问题对应的割平面和对偶线性规划之解，都在公开数据之列。而计算机程序比较精致，包括6646行C语言代码，能够遍历所有子问题，证明对偶线性规划问题之解可以给出界并用来逐个舍弃子问题。这样的证明不如毕达哥拉斯定理[2]的证明那么简洁，但是确实为未来的TSP研究者留下了充足、丰富的信息，让后来者能够追根究底，真正理解这场大规模计算的成果。

1. Applegate, D. L., et al. 2009. Op. Res. Let. 37, 11-15.
2. 又称勾股定理。——译者注

8.2 规模宏大的TSP

TSP的计算之美在于一个简单的事实：题目规模没有最大，只有更大。

8.2.1 Bosch的艺术收藏品

告别TSPLIB测试题集，就轮到Bob Bosch的挑战题目登场。我非常喜欢他创造的这些难题，编写题目所用的技术将在第11章介绍。他的六道题目规模各异，最小的是100 000座城市的“蒙娜丽莎”TSP，最大的则是200 000座城市的维米尔名画《带珍珠耳环的少女》“摹本”。数据集公开在网络上，任何人如果有意寻找更好的路线，或者确定路线长度的下界，都可以自由获取。[1]

1. 参见http://www.tsp.gatech.edu/data/art/index.html。

Bosch测试问题中，“蒙娜丽莎”TSP迄今吸引的注意力最多，毕竟它只比85 900座城市的求解纪录高出一丁点儿，难免让人心痒手痒。可是，85 900座城市和100 000座城市之间差别虽小，或许却是障眼法。无论怎么看，“蒙娜丽莎”TSP都很可能比当前的世界纪录要难得多。实际上，后一道计算机电路实例如图1-8所示，可以看到，其实有许多城市紧密排成直线。这种几何性质表明，那道85 900座城市的题目也许外强中干，规模大但难度却相对较低。再看看“蒙娜丽莎”TSP的各点分布，简直是要多复杂就有多复杂。

第1章已经提过，永田裕一在2009年3月17日发现了“蒙娜丽莎”TSP的当前最好路线，谁能找到更短的路线，就能获得1000美元的奖金。2009年2月和3月，Bosch刚刚创建完成数据集，全世界最优秀的路线寻找专家纷纷活跃起来，永田裕一的计算结果将竞争推向高潮。在此期间，最好路线的纪录六易其手，他凭借长度为5 757 191的路线拔得头筹，这比前一天由Keld Helsgaun发现的路线短了8个单位长度。但是这就是最优路线吗？

2010年1月18日，5 757 044确定为“蒙娜丽莎”TSP路线的下界。至此，下界与永田裕一路线之间仅仅相差147个单位长度。看上去近在咫尺，只有0.0026%而已，可是差距依然存在。新的下界是由Concorde分支切割搜索得到的，总共历经1065个子问题，耗时66天，换算成计算机

时间则长达4.37年。也许继续运行几轮Concorde程序，有机会再缩减几个单位长度的差距。但是要想真正填补缺口得出结论，大概必然需要新的思想，尤其是割平面分离的新思路。对我这种计算狂人来说，现在好戏才刚刚开始呢。

8.2.2 世界

世界旅行商问题有1 904 711座城市，正在等待有识之士前来做出TSP计算的一流突破。它的数据取自美国国家图像与测绘局（National Imagery and Mapping Agency）和美国地质调查局（Geographic Names Information System）数据库。该题于2001年问世，覆盖了当时全球各地人类居住的每一处。这些位置由经度和纬度指定，旅行成本则是把地球视为球体，用两点间的大圆劣弧长度近似给定。如此定义的成本函数是由TSPLIB里的GEO标准稍加变化而得到的，GEO标准以千米为单位，而这里则换算为米作单位。

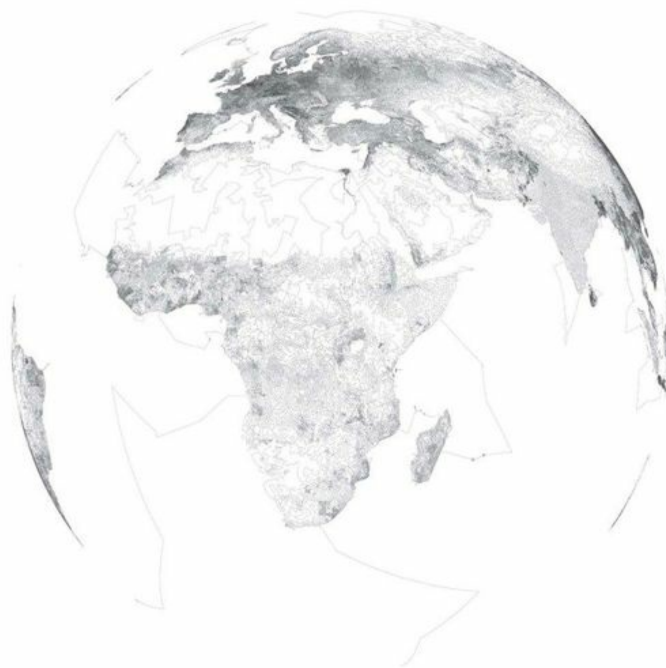


图**8-11**　周游世界所有城市的路线，图片由**David Applegate**提供

2001年秋季，研究者根据一条长度为7 539 742 312米的初始路线，以及数值为7 504 218 236的路线下界，确定两者之间有0.47%的最优性差距（optimality gap）。从那时起的10年间，新的计算结果不断地“蚕

食”上述差距，路线改进方面主要归功于Keld Helsgaun，而下界加强方面则基本归功于Concorde程序。最优性差距不断缩小，如图8-12的柱状图所示，其中下界加强带来的改进用红色条块表示，路线优化带来的改进则用灰色条块表示。当前的最好路线是Helsguan提出的，长度为7 515 790 345，最强的线性规划下界则是Concorde程序给出的7 512 218 268，由此可得，差距为0.0476%。

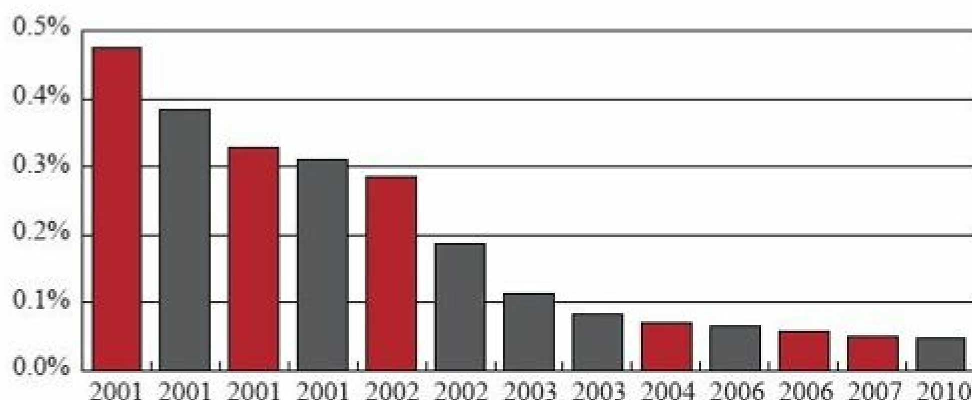


图**8-12**　世界旅行商问题的最优性差距逐年递减

过去10年间，最优性差距不断压缩，如今只有原始值的1/10强，可谓进步显著。有趣的是，其中近乎75%的改进都是由路线优化获得的。在2001年的TSP世界，谁也没有预料到上述事实。当时人们普遍认为，TSP启发式算法是尖端技术，能给出近优解，所以肯定是线性规划下界太弱才导致了一开始的最优性差距。Helsgaun的LKH程序表明，寻找路线的方法还有改进空间，从而改变了传统的观点。此时此刻，我无法猜出最优值究竟位于何处，不知道它究竟是距离路线长度7 515 796 609更近，还是距离下界7 512 218 268更近。

尽管如此，对Concorde程序加以工程改进，肯定能够提升线性规划的界。比如，目前由于数据集规模太大，无法在程序中使用多米诺奇偶性分离模块。但这其实是个好消息，说明我们又有活儿可干了！

8.2.3 恒星

2003年，我和Dave Applegate设计了一道TSP题目，囊括了美国海军天文台USNO-A2.0星表（United States Naval Observatory A2.0 catalog）里记录的全部天体，共计有526 280 881个之多。至少在可以预见的未来，这或许就是TSP计算的终点站。我们最初想要收入每对天体之间的距离估计值，如果能让“进取号”（Enterprise）在下一个五年任务里沿着

最优路线前进，将会很有意思。[1]但是遗憾的是，数据太粗糙了，会导致所有恒星都落在为数不多的几个同心球面上。因此，我们决定改为模拟望远镜的移动方式。这样一来，数据就把TSP城市指定为天空中的位置，任意一对城市之间的旅行成本便可由两点确定的夹角度数给定。

1. “进取号”是著名科幻系列作品《星际迷航》（又译《星际旅行》）中的星际舰船，它所肩负的五年任务是探索新世界，寻找新生命及新文明，驶向前人未至之境。——译者注

整体处理恒星旅行商问题非常困难。当n取值高达5亿时，即使是n^2复杂度的运行时间也太漫长了。故而，目前的目标就是发展拆分数据集的方法，确定各部分的界和路线，然后再把它们拼到一起。按照这种方法，我和Dave Applegate、Keld Helsgaun、Andre Rohe做了初步的试验，并于2007年获得了0.41%的最优性差距。这意味着，恒星旅行商问题的状态落后于世界旅行商问题，相差大约10年的光景。所以，如果你对路线或者界限有什么奇思妙想，请记得用在这个大规模数据集上。



图**8-13**　星图（由美国宇航局戈达德太空飞行中心科学可视化工作室提供）

第9章 复杂性

> 对于显然具有限界的整数规划问题，多项式时间算法是存在，还是不存在？我的意思是，这就是我的演说：是存在，还是不存在？
>
> ——Jack Edmonds，1991年[1]

1. Edmonds (1991).

随着数字不断增长，追踪旅行商的研究带来了数学、计算和工程领域的突破，也推动了无数实际应用的进展。对TSP研究者而言，这就是骄傲和快乐的源泉。可惜，一步一步推进研究的方法并没有回答最为根本的复杂性问题：我们究竟能否高效解决每一道TSP题目呢？

从这种复杂性的观点看来，通过Stephen Cook和Richard Karp的理论，旅行商的命运能与其他许多问题的命运联系起来，比如一般整数规划问题。事实上，TSP隶属于P与NP问题，而P与NP问题则跻身于七道“千禧年大奖难题”（7 Millennium Problems）之列。克雷数学研究所为每一道难题悬赏百万美金。在其官方网站上，P与NP问题的介绍如下。

> 如果一个解是否是一道题目的正确解很容易检验，那么求解这道题目是否同样容易？这就是P与NP问题的本质。NP问题的典型范例是哈密顿通路问题（Hamilton Path Problem）：给定*N*座需要访问的城市（开车前往），如何访问所有城市而不重复抵达任何一座城市？给我一个解，我能轻松验证它的正确性，但是我无法如此轻松地根据我已知的方法找到一个解。

P与NP问题常常如此公开描述，使用TSP或其改编形式来激发读者的积极性。本章中，我们会讨论旅行商问题的一般复杂性，介绍这方面的已知事实和未知空白。

9.1 计算模型

数学命题必须清晰准确才能有意义，至少必须能够改写成清晰准确的表述形式。百万美金的复杂性问题也不例外。我们必须明确所谓算法有什么意义，换言之，“可计算的”是什么意思。20世纪伊始，David Hilbert提出可判定性问题（Entscheidungsproblem），引起了人们对此的关注。可判定性问题大体相当于询问，是否存在一个算法，对任意给定命题都能判定它是否可以根据一组公理证明。解决此类问题的理论不断发展，成为20世纪数学的一项美妙成就，开路先锋是大师级人物Kurt Gödel（克尔特·哥德尔）、Alonzo Church（阿隆佐·邱奇）和Alan Turing（阿兰·图灵）。

算法的直观概念就是一系列合起来能够求解某问题的简单步骤。约在距今2300年前，欧几里得提供了计算最大公因数的算法。但直到Hilbert的年代，人们仍不清楚，如何才能给出算法的一般定义。在1936年那篇著名的论文中，图灵给出了一个答案，并引入了一类数学模型，称为图灵机（Turing machine）。[1]

1.Turing, A. M. 1936. Proc. Lond. Math. Soc. 42, 230–265.

图灵机有一条带子，用来存放符号；有一个读写头沿着带子移动，用来在各个存储单元里读取及写入符号；还有一个控制器，用来引导读写头移动。图灵机有一组数目有穷的状态，其中有两种特殊状态，分别是初始（initial）和停机（halt）。控制器实际上是一张表，指示处于特定状态s的机器在读入特定符号x时应该执行什么操作。具体说来，应该执行的操作有：在带单元内打印新符号x'，将读写头向左或向右移动一个单元，进入新状态s'。要想用图灵机解决问题，机器首先在初始状态启动，输入写在带子上，到达停机状态时便终止运行。

为了趣味性，可以考虑一台真实的图灵机，它的读写头沿着一条窄窄的长带子移动，带上写满了符号。事实上，有些学生改造了数个鞋盒，在每个鞋盒末端固定了一卷纸带，又在便笺本上记录下来状态之间的转移情况，把成品的照片传到了网上。图灵本人并没有提到真实的机器，而只是在论文中举了几个例子，强调说明了一个事实：写下一台机器的状态转移表，就可以完整地描述这台机器。

下面让我们考虑一个简单的例子：给定一串由0和1组成的数字串，试确定1的个数是奇数还是偶数。为了解决此问题，构造的图灵机可以有初始（initial）、奇数（odd）、偶数（even）、停机（halt）这四种

状态，有0和1这两种符号，转移表如图9-1所示。表中，每行对应一个符号（包括空格“_”），每列对应停机以外的一种状态。每个表项都是三元序列，指明要写的符号，读写头在带上移动的方向，以及下一个状态。比如，机器如果处于奇数状态，并且读到符号1，那么就在单元内写入一个空格符号，向右移动一个单元，并改为偶数状态。如果给定由0和1组成的数字串，数字位于带子上的连续单元内，读写头位于最左端的符号处，那么我们的图灵机就会向右进行操作，直到遇到空单元才停止。空单元是串的结束标志。图灵机停机时，若1有偶数个，则在带上写入0；反之，若1有奇数个，则在带上写入1。过程简单明了，用转移表就说明了操作的概念。

	初始	奇数	偶数
0	_,向右,偶数	_,向右,奇数	_,向右,偶数
1	_,向右,奇数	_,向右,偶数	_,向右,奇数
_	0,_,停机	1,_,停机	0,_,停机

图**9-1**　奇偶性校验图灵机的转移表

下一步可以构建一台加法图灵机，对二进制表示的两个数求和。计算复杂性方面的大学课程中，这是一道常见的练习题，而且在这类题里面还算是挺有意思的。如果你想更加熟悉机器操作，我建议你也试试这道题。你会注意到，附加一条带子用来存储中间计算结果会很方便。所以，可以很自然地定义这一类图灵机，多带图灵机有多条带子，每条带都有单独的读写头。虽然附加带子能带来便利，但是不管想要计算什么，只要多带图灵机能做到，那么单带图灵机也一样能做到，只不过速度有点慢而已。

上面最后那句话的意思是说，单带图灵机可以模拟多带图灵机。这很重要。我们希望把算法定义成在单带图灵机上可执行的东西，不过这一条定义是否足以体现出我们对算法的一切要求呢？对此，只能回答：迄今为止，图灵机一直能够见招拆招，应付一切任务。某件事如果在现代计算机上是可计算的，那么迅如闪电的图灵机也能完成这一计算。

所以，我们可以把算法与图灵机等同看待，这种有效的假定称为邱奇-图灵论题（Church-Turing Thesis）。[2]该论题广受认可，它给出了算法的正式模型，使P与NP问题及其他复杂性问题得以准确表述。也许有朝一日，我们会遇到异乎寻常的计算能力，从而考虑对算法的定义进行扩充。不过，过去七十余年里，图灵提供的定义恰好足够满足研究界的

需求。

2. 邱奇与图灵几乎同时完成研究。他用一种不同的框架结构描述了可计算性，后来图灵证明，这种描述和图灵机的概念实际是等价的。

通用图灵机

举个例子，像iPhone这样的现代电话和我们脚上穿的鞋子之间有着根本性的区别。鞋子只是针对保护双脚的单一功能而设计的，但iPhone手机有数十万应用可供挑选，设计其硬件时想象不到的任务也能执行。我们把这当成是理所当然的事情，然而，发明可编程机器的这步飞跃却是充满智慧的创举，这正是图灵在那篇论文中提出的构想。

图灵机是用来描述算法意义的强大模型，但是一种图灵机只是为了一个任务而设计的，比如对两个数求和。按照这种意义理解，图灵机更像是鞋子而不是iPhone。不过图灵指出了至关重要的一点：可以设计通用图灵机（Universal Turing machine），它能够模拟每一种图灵机。

> 可以发明一台机器，它能用来计算任何可计算序列。如果向这台机器U提供一条带子，带子起始端写有某种计算机器M的S.D.，那么U就能像M一样计算同一段序列。

这段引文中的“S.D.”是“标准描述”（standard description）的缩写，图灵用这一术语来称呼转移表。因此，上文的意思是说，要在带输入中包含一个转移表，就像在现代计算机中包含一个程序一样。图灵的想法，再加上Konrad Zuse和约翰·冯·诺依曼等人的辛勤工作，共同开启了计算时代。

9.2 Jack Edmonds的奋战

David Hilbert呼吁提出算法的理论，而图灵出色地作出了回答。然而，自从数字计算机问世以后，效率问题便很快成为最基本、最重要的问题。知道一个问题能由图灵机求解是一码事，但是知道能在有生之年看到图灵机给出问题之解就完全是另一码事了。

有关算法效率的早期讨论围绕TSP等整数规划模型展开。下面这段话颇能代表这段时期的讨论情况。它摘自一篇1953年的论文，作者是Martin Beckmann和诺贝尔奖得主Tjalling Koopmans。[1]

1. Beckmann, M., T. C. Koopmans. 1953. Cowles Commission: Econ. No. 2071.

> 应该加上一点，在所有讨论的指派问题中，当然存在显而易见的暴力求解法，即枚举所有指派方案，分别计算最大化目标的值，选出给出最大值的指派方案。对于大多数具有实际意义的情形，这种方法的成本都太高。而所谓解法，指的是能把更多情形的计算工作量降低到可控范围的求解过程。

Beckmann和Koopmans考虑了一组问题，其中既有TSP，也有将若干工作分配给若干工人的标准指派问题。次年，即1954年，Merrill Flood为高效解法提出了支持的理由。[2]

2. Flood (1954).

> 有传言说海军在搭建一台计算机，以解决各种形式的油轮调度问题。这台计算机将带来重大优势，不在于节约执行计算的成本，而在于缩短对运算进行验算所需的时间。这一点，怎么强调都不过分……使用高速计算机可能确实会多花一些钱，但是会使计算具有时间上的可行性。

Flood考虑到这种速度需求，继续评论道，对于TSP，“迄今没有可接受的计算方法”。计算的有穷性不够令人满意，那么判定算法品质时，我们的目标是什么呢？虽然这个问题亟待解决，可是在20世纪50年代却没有出现明确的答案。

现在，轮到Jack Edmonds登场了。我们在本节标题里用了“奋战”一词，没错，就是这个意思——当时人们普遍认为，优于有穷的事情就不该归数学界解决了，而Edmonds不得不与这种舆论作战。他本人在1954年说过：“对于理论数学家而言，利用现有计算机器求出切实可用的解，往往并不是个有趣的问题。”这正是困难所在。诚然，Flood、

Koopmans、Kuhn等人对实用求解方法产生了兴趣，单纯形算法在求解线性规划问题方面取得了惊人的成功，可是这或许也妨碍了人们直接讨论优于有穷的算法。之所以造成这种麻烦，是因为单纯形算法似乎能够解决遇到的一切线性规划问题，虽然未能证明它每次都会高效运行。这就导致，算法即使没有性能保证，也能得到人们过分放心的认可。



图**9-2　1964**年美国国家标准局研讨会，右一为**Jack Edmonds**，照片由**William Pulleyblank**提供

Edmonds当时面临的工作非常困难。他的游说活动始于1961年夏天的兰德公司，那一年，他和一群年轻研究者受邀参加研讨会，与会人士还包括该领域的重要人物，比如Dantzig、Fulkerson、Hoffman，等等。Edmonds在兰德公司作了报告，讨论了寻找图中最优匹配的问题。研讨会期间，他成功提出了该问题的一个好算法，换言之，算法步骤数的增长速率至多正比于n^4，其中n是图中顶点的个数。这个深刻的结果成为了Edmonds的奋战重点，他的数学运算非常美妙，动摇了他人的观点。但是，Edmonds一路上也经历过许多艰难险阻。他有一篇非常值得阅读的回忆录，其中写有下面这段话。[3]

3. Edmonds (1991).

> 当年，我在大肆宣扬这件事的时候——我还记得自己着了魔，不遗余力地谈论它——遇到的最大反应就是：“呃，指望出现这种情况有点傻吧，而且我看它好像没什么实际意义，哦对了，如果是n的28次方怎么办，你知道的，那就不……”诸如此类。

今天，没有人质疑他的理论。Edmonds成为了算法和计算复杂性领域的英雄领袖。

有一样东西没有沿用下来，就是“好”这个词的用法。当时，如果一个算法保证能在至多正比于n^k的时间内完成，就称之为好算法，这里n是问题规模的量度，k是某个固定的幂次。第1章已经提到，如今的标准术语是“多项式时间算法”。这可能是个不错的改动，毕竟像单纯形方法

这么成功的算法，如果也要说成坏算法，实在有些残酷，让人于心不忍。

9.3 Cook定理和Karp问题列表

计算复杂性早期的历史一日千里。1967年，Edmonds刚刚在匹配等组合问题上取得成功，就提出了惊世骇俗的猜想，认为TSP也许根本没有好算法。他的多面体方法对于其他情形都有卓越的效果，他又为何不看好它在旅行商问题上的应用呢？Edmonds含糊其词，不愿解释，只是指出不存在好算法的可能性是合理的。4年之后，Stephen Cook和Richard Karp把这个问题纳入了P与NP问题的大背景中，创立了他们的理论。

9.3.1 复杂性类

数学家喜欢让一切都井井有条。对于复杂性理论而言，这就意味着他们重视判定问题（decision problem），即答案为“是”或“否”的问题。比如，某个图里是不是存在哈密顿回路？要么回答“是”，要么回答“否”。再比如，给定一组城市，是不是存在一条短于1000英里的周游路线？要么回答“是”，要么回答“否”。[1]

1. 判定问题未必直接体现TSP本质，但是我们总可以提出一系列答案为“是”或“否”的问题，找出最短问题序列对应的最短路线长度。

Richard Karp发明了缩写记号P，用来表示有好算法的判定问题。P类的正式定义是指，能在多项式时间内由一台单带图灵机解决的问题类，换言之，如果输入带上的符号数目为n，那么必定存在指数k和常数C，保证图灵机经过至多为Cn^k步以后必然停机。当然，这种定义相当好，单带图灵机可以替换为多带图灵机，甚至换成功能强大的现代数字计算机，也不会影响问题分类。诚然，用图灵机模拟先进计算机会拖慢计算速度，但是速度放慢的系数关于n仍然只是多项式关系。所以，我们如果有先进计算机上的多项式时间算法，就同样拥有单带图灵机上的多项式时间算法。

对判定问题而言，属于P类就意味着完美。不过，Stephen Cook研究了另一类自然出现的问题，其范围或许比P类更大。他概述了Edmonds的看法，考察了能够在多项式时间内验证肯定答案的问题。要想验证答案是否正确，我们同时给出问题叙述和肯定证据，让图灵机检验答案是否确实为肯定的。例如，要想验证一组城市能在1000英里以内周游一

遍，提供一条短于1000英里的周游路线作为证据即可。[2]

2. 这类可在多项式时间内验证问题的概念也曾由Leonid Levin独立提出。

借助非确定性图灵机（nondeterministic Turing machine），可以用另一种方式看待上述证据。非确定性图灵机在计算时能够复制自身，因此它不是真实存在的。如果存在多项式时间的验证算法，那么一台非确定性图灵机就可以创造出自身的多个副本，其中一个副本可以猜中肯定证据，从而确定原问题的答案是肯定的。基于这种看法，Karp提出，可以把Cook研究的问题类简称为NP。

乍一看，似乎NP类远没有P类那么严格，把问题划分为NP类比划分为P类容易得多。TSP就是一例，检验解答容易，找出解答或许很难。再举一个例子，考虑因数分解问题，也就是把给定整数写成两个更小的整数之积。分解过程可能很难（尚未找到多项式时间算法），而要检验某个答案是否正确就是小菜一碟。类似的例子可以构造出很多，不过，它们只能暗示NP类可能比P类范围更大，实际上，对于只属于NP类而不属于P类的问题，我们居然一个也没发现。

9.3.2 问题归约

Stephen Cook的论文开了NP问题正式研究之先河。文中，他提出一道逻辑题可能不属于P类。“另外，上述定理表明，{重言式}（tautology）很可能属于一组不在L*内的有趣问题，我认为这一猜想值得花一番工夫尽力解决。若能给出证明，必将取得复杂性理论的重大突破。”[1]事实上，如今，这个猜想的证明价值百万美金。Cook发表论文时，Karp尚未建立起如今通用的标准术语——Cook所谓的L*如今改称为P，而所谓{重言式}如今则通称为可满足性问题（satisfiability problem）。可满足性问题要求确定，对于一系列逻辑变量，能否赋值为真或假，使得给定的表达式取值为真。表达式中用到的量都是这些逻辑变量及其否定（“非”），联结词则为逻辑“且”和逻辑“或”。不过，Cook提出猜想的根据比该问题本身更加重要，因为他的定理表明，每个NP问题都可以表述为可满足性问题。

1. Cook, S. 1971. In: *Proceedings of the 3rd Annual ACM Symposium on the Theory of Computing*. ACM Press, New York. 151–58.

Cook理论的核心思想是归约，即把问题简而化之。本书中已经多次用到归约思想，例如Karl Menger在维也纳研讨会上曾经提出的问题：寻找经过给定一组点的最短路线。这并不完全等同于TSP，因为并没有要

求终点和起点重合。但是，假如我们会求解TSP，那么只需加入一座虚城市，令它与其他给定点之间的旅行成本均为0，就可以依样解决Menger的问题。

问题归约（problem reduction）的正式定义为，在多项式时间内，图灵机接受问题A的任意实例并据此构造出问题B的实例，使得A和B的答案完全相同，要么都为“是”，要么都为“否”。在把Menger问题归约为TSP的时候，我们额外加入了一座城市和*n*个距离数据，所以，可以设计一台图灵机在正比于*n*的步骤内实现问题归约，这里*n*就是原问题给定的点的数目。问题归约大抵如此，你只需牢记一点，问题B的规模不应当过分超出A的规模。

显而易见，在对诸多NP问题进行分类时，归约用途不小。要说明某问题很简单，可以试试把它归约为另一个简单的问题；要说明某问题很难，可以试试把另一道已知的难题归约为它。归约能带来意想不到的有序度：经Cook证明，每个NP问题都可以归约为可满足性问题。

存在大一统问题的理念非常深刻，极其犀利，不过Cook定理的证明其实并没那么难。首先，注意到检验NP问题只需要多项式量级的图灵机步骤，所以把NP问题归约为可满足性问题时，可以在每一步检验步骤里加入标志机器状态和读入符号的逻辑变量。由此便可给出证明，具体细节这里不准备详述，我只想要指出，Cook原始论文里的完整证明也只用了不到一页纸，虽然字号挺小。

现在，我们可以理解Cook的推理过程。从问题A能归约到问题B，蕴涵结论“若B属于P类则A亦属于P类”。因此，如果可满足性问题属于P类，那么所有NP问题都存在多项式时间算法。Cook认为P=NP的可能性不大，由此便提出了上述猜想。

9.3.3 21个NP完全问题

把可满足性问题称为“大一统问题”虽然符合Cook的结论，但是无法全面体现出他的问题归约理论的重要意义。事实上，知道可满足性问题统领NP类，就可以直接证明其他问题同样具有这一性质。

如果某个NP问题可由所有NP问题归约得到，那么它就称为NP完全问题。Cook在证明后写道，可满足性是NP完全的。他的论证很简短，只是指出图论中的子图同构问题同样也是NP完全问题，又证明了可满足性问题可以归约为子图同构问题。因此，NP问题都可以先归约为可满足性问题，再归约为子图同构问题，可以设计一台图灵机，依次执行

这两步问题归约，这就证明了子图同构问题是NP完全的。

由于这种连锁式问题归约的思想，复杂性理论领域变得热火朝天，研究领军人物是Richard Karp，他的论文[1]写于Cook发表结论一年以后。Karp漂亮地给出了P类、NP类、图灵机和问题归约的专业阐述，还提出了如今赫赫有名的21个NP完全问题列表，并列出了Cook可满足性定理给出的归约。列表中，TSP以两种不同面目出现，分别是无向图的哈密顿回路问题和有向图的哈密顿回路问题。

1. Karp (1972).

Karp的论文面世以后，传遍了大街小巷，其他难题的归约也跟风出现，比比皆是，成百上千道问题获证为NP完全问题。1979年，Michael Garey和David Johnson出版了具有里程碑意义的著作，书名为《计算机和难解问题：NP完全导论》（*Computers and Intractability: A Guide to the Theory of NP-Completeness*），在算法研究圈子里几乎人手一本。每次遇到新问题，研究者总是会先查一遍Garey-Johnson的NP完全问题名录，看看问题归约是否有可用的参照。[2]

2. Garey和Johnson（1979）极好地介绍了计算复杂性理论。近来，Arora和Barack（2009）也合著了一本优秀的导论。

9.3.4 百万美金

在实践中，一旦证实一个问题为NP完全问题，研究者便会认为它很棘手，不好解决，于是要么使用方便但粗糙的启发式算法，要么使用TSP求解中涌现出的某种辛苦繁重的方法。他们依据的假设是，NP完全问题不可能有高效的多项式时间算法。但是迄今为止，并没有充分靠谱的证据支持P和NP不是一回事。那么，请问本书读者，你支持哪一方观点呢？你认为P=NP，还是P≠NP呢？

如今，计算领域的重要性与日俱增，P与NP问题或许已因此成为整个数学界最知名的未解难题。尽管不断有人做出求解的努力，但是在2009年，Lance Fortnow综述该问题的研究现状时，依然以四个字作结：尚未解决。不过，毕竟有价值百万美金的克雷大奖，我们有理由期待在不久后就将迎来新的进展，正如Wowbagger the Infinitely Prolonged当初的那句台词一样——在道格拉斯•亚当斯的《银河系搭车客指南》（*The Hitchiker's Guide to the Galaxy*）中，Wowbagger the Infinitely Prolonged打算挑衅宇宙中的每个男人和每个女人，当这项TSP计划的可行性遭到质疑的时候，他回答道：“人总是可以有梦想的，是吧？”

9.4 TSP研究现状

Gerhard Woeginger在荷兰埃因霍芬理工大学工作，他也是克雷大奖的民间档案管理员，保管无数大胆的申领文件。他的“P与NP”网页的亮点在于，上面按照时间顺序列出了该问题的里程碑式成果，并在每条成果开始相应注明“相等”或“不相等”，例如[1]。

1. 网站经过更新，2013年初，上面的序号44~46已变为47~49，特此说明。——译者注

> 1. [Not equal]: In September 2008, J. J. proved . . .
> 2. [Not equal]: In October 2008, S. T. established . . .
> 3. [Equal]: In November 2008, Z. A. proved . . .

翻译过来就是。

> 1. [不相等]：2008年9月，J. J.证明了……
> 2. [不相等]：2008年10月，S. T.证明了……
> 3. [相等]：2008年11月，Z. A.证明了……

Woeginger提供了每篇论文的链接，有些时候还给出了反驳意见的链接。[2]有25篇论文支持“P=NP”，24篇论文支持“P≠NP”，双方可谓势均力敌，不相上下。读来有趣，可惜迄今为止，每篇论文都在严肃的审查面前败下阵来。

2. 参见http://www.win.tue.nl/~gwoegi/P–versus-NP.htm。

Woeginger的列表中，25条“相等”结果都是通过设计TSP变体问题的好算法得证的。这些方法五花八门，既有简单的枚举法，也有人不辞繁复，试图针对旅行商问题，构造出多项式时间的完整的线性规划方案。它们都有严重的漏洞，无一例外。不过，求解旅行商问题确实是证明P=NP的有望途径。

9.4.1 哈密顿回路

如果你准备向百万美元的大奖迈进，而你的计划中有一步是搜寻多项式时间的TSP算法，那么有必要牢记在心：只考虑有限定条件的TSP，足矣。很多人会选择研究哈密顿回路的判定问题，探究图中是否

存在哈密顿回路的判定方法。我们知道，这道题是TSP的变体，也属于NP完全问题。不仅如此，如果假定输入的图是二分图（bipartite），该问题依然是NP完全问题。所谓二分图，即顶点可以分别染色为红蓝两色，使得每条边的两个顶点都为一红一蓝。具有如上限制的特殊结构可以用在算法中，但是要论多项式时间内解决的可能性，哈密顿回路问题其实不比完整TSP简单。

在此基础上，Alon Itai、Christos Papadimitriou和Jayme Szwarcfiter思考得更加深入。他们证明，可以具体说明哈密顿回路问题，把考虑的图限定为网格图（grid graph），即无穷网格的有穷子集。取一大片矩形网格，删去顶点的一个子集，就得到网格图，如图9-3所示。网格图为回路搜索算法提供了诱人的NP完全问题目标。David Johnson和Christos Papadimitriou称之为“TSP的终极特例”。[1]

1. 见Lawler (1985)，“Computational Complexity”（计算复杂性）章节。

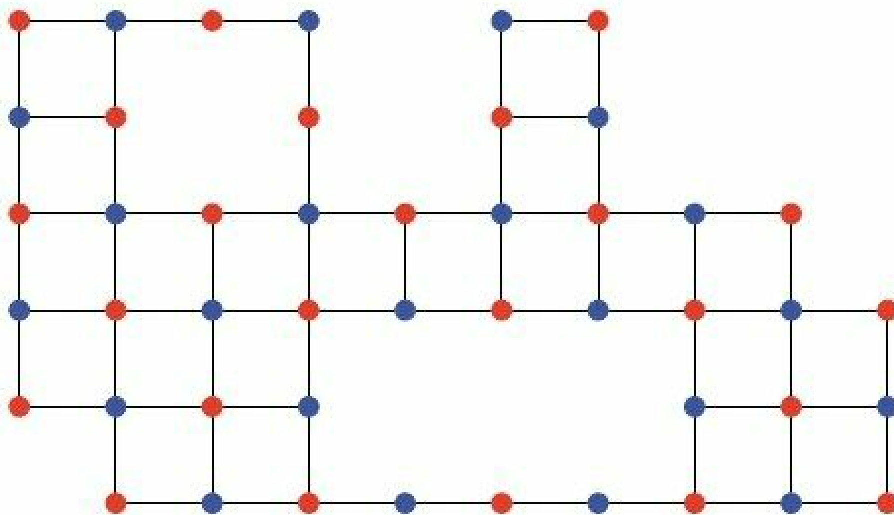


图**9-3**　网格图

9.4.2 几何问题

哈密顿回路问题清楚明白，没有枝节，不需要劳心计算旅行成本。可惜，对于我们多数人来说，为方便直观理解，TSP题目最好还是加以限定，使用欧几里得距离，以(x, y)坐标指明城市位置，以城市间直线距离作为旅行成本。解决这种TSP题目也能证明P=NP，收获百万美元的大奖。[1]

1. 为理解该结论，注意到网格图上的哈密顿回路问题可以作为欧氏距离TSP求解。

除了大奖难题以外，在同一背景下，还有另一个重要的问题悬而未决，而且乍一看令人意外：目前尚不知道，使用欧氏距离的TSP是否确实属于NP类。改用判定问题的方式来叙述，也就是问，指定数字K，是否存在长度至多为K的路线。因此，按照满足要求的路线顺序，列出所有城市，就构成了自然的证明。为简化数据显示问题，可以假定(x, y)坐标取值均为整数，则技术上的困难就在于计算平方根。近似计算每一个数的平方根都很简单，想算多少位都能轻松算出多少位，可是，把它们加起来就麻烦了，它们的和很可能距离K只差一丁点儿。要确定路线长度真的不大于K，就要算出足够精确的近似值。能否在多项式时间内做到这一点？目前尚无定论。

20世纪80年代初期，Ronald Graham普及了这道平方根之和问题。他举例说明了该题的困难之处：下面有两行数，每个都加上1 000 000，然后分别取它们的平方根之和。

1 25 31 84 87 134 158 182 198

2 18 42 66 113 116 169 175 199

例如，由第一行得到的和是√1000001+ √1000025 +. . .+ √1000198。看起来平淡无奇，可是算完就会发现，得到的两个值分别为

9000.4499835688397309490268288613590291912

9000.4499835688397309490268288613590291915

两者直到小数点后第37位才首次出现差别！一般说来，目前仍不清楚，要确定一系列平方根之和是否不大于给定数K，多项式量级的数字位数是否够用。具体到欧氏距离TSP的例子里，输入的规模是K和各个(x, y)坐标的数字位数。

解决平方根之和问题，便可长驱直入，直接证明欧氏距离TSP属于NP问题。不过，要证明一组点有长度至多为K的周游路线，这不是唯一途径，或许还有其他方式，只不过需要借助尚不为人所知的几何结构。这方面的研究很有意思。至少就TSP而言，这也许比平方根之和所需的数论方法更值得研究。

9.4.3 Held-Karp纪录

要想确定旅行商问题在多项式时间内能否解决，不管是给出肯定答案还是否定答案，大概都需要有颠覆性的革新思想才能做到。不过，另一个目标没那么困难：反复改进TSP算法最知名的运行时间界限，一点

一点瓦解问题的复杂性。这种逐步方法自有其诱人之处，毕竟如果算法能获得更快的时间界限，就可能适用于实际应用，从而推进TSP的实战前线。

“没有最快，只有更快”确实是复杂性分析的座右铭。但是在TSP领域，似乎早在1962年，研究者就遇到了无法翻越的高峰——Michael Held和Richard Karp的研究成果[1]。他们的动态规划算法能在正比于$n^2 2^n$的时间内，解决任何一道n座城市的TSP题目。时至50年后的今日，境况依然如此。要超越Held-Karp纪录，或许不需要“革命”那么夸张，但显然需要令人眼前一亮的创新。

1. Held和Karp (1962).

鉴于Held和Karp是最高纪录保持者，不完整介绍一下他们的算法，实在是说不过去。为方便解说，考虑n座城市的TSP题目，城市从1到n依次编号命名，每两座城市之间的旅行成本记为*cost*(1, 2), *cost*(1, 3), 等等。

固定城市1作为旅行社的出发地，Held-Karp解法根据最优子通路得到解，而最优子通路则分别针对不包含城市1的每个城市子集和其中每个终点得出。以子集{2, 3, 4, 5, 6}为例，若选取6为终点，则所谓最优子通路需要满足：从城市1出发，到城市6结束，按任意顺序经过城市2、3、4、5，并且是所有此类通路中成本最低的。这样一条最优子通路的旅行成本记为*trip*({2, 3, 4, 5, 6}, 6)。为计算该值，求和式

$$trip(\{2, 3, 4, 5\}, 2) + cost(2,6)$$
$$trip(\{2, 3, 4, 5\}, 3) + cost(3,6)$$
$$trip(\{2, 3, 4, 5\}, 4) + cost(4,6)$$
$$trip(\{2, 3, 4, 5\}, 5) + cost(5,6)$$

中的最小值，这四个和分别对应子通路中倒数第二座城市的四种选择方式，即转化为首先按最优路线前往倒数第二座城市，然后再从那里出发前往城市6。

从若干个关于四座城市的数值出发，构造关于五座城市的trip值，正是Held-Karp方法的核心所在。该算法如下进行。第一步，计算所有关于一座城市的数值。这很容易，例如trip({2}, 2)就是cost(1, 2)而已。第二步，使用关于一座城市的数值，计算所有关于两座城市的数值。第三步，使用关于两座城市的数值，再计算所有关于三座城市的数值。以此类推，最后得到(n-1)座城市的数值时，即可直接读出最优路线的成本，因为在和式

$$trip(\{2,3,...,n\}, 2) + cost(2,1)$$
$$trip(\{2,3,...,n\}, 2) + cost(2,1)$$

$$trip(\{2,3,...,n\},3)+cost(3,1)$$
$$\ldots$$
$$trip(\{2,3,...,n\},n)+cost(n,1)$$

中，cost项对应于返回城市1的路程，所以最小值就是所求结果。

原理就这么简单，介绍完了。下面推算运行时间界限。首先，在*n*座城市的题目里，有2^{n-1}个不包含起点的城市子集，这是基本事实。在每一个子集里，终点至多有*n*种取法（事实上，取法数目只是相应子集的基数，但是为方便计算，此处统一增加到最大值*n*），trip值计算过程包括少于*n*次加法和少于*n*次比较大小。2^{n-1}乘以*n*再乘以2n，可知总步骤数不会超过$n^2 2^n$。

只要城市数目*n*超过10，那么上述运行时间界限就比枚举检验所有路线更好。但假如这个Keld-Harp结果就是最好结果，难免让人灰心丧气。在尝试突破纪录时，挑战者应当把重点放在2^n项上。即使把$n^2 2^n$改进为n^{2n}，也不能算是重要提升；但一旦改进为$n^2(1.99)^n$或者$n^2 2^{\sqrt{n}}$，就堪称重大新闻，甚至可能标志着新时代的开端，其后的新改进可能带来更好的实用方法，具有强有力的运行时间保证。[2]

2. 此类研究的优秀参考资料见：Woeginger, G. J. 2003. Lect. Notes Comp. Sci. 2570, 185-207.

9.4.4 割平面

若想打败Held-Karp，何不试试割平面法？Randall Munroe是xkcd.com的作者（画手）兼创始人，第1章的图1-5就是出自他笔下的TSP漫画。在发布该画作时，他的评论正中要害："最好的线性规划割平面方法，属于哪个复杂度类呢？我遍寻而不得答案。嗨，画加菲猫的那哥们儿可不会碰到这种问题。"[1]

1. 请访问Randall Munroe的网站xkcd.com，在编号为399的TSP漫画上方悬停鼠标，就能看到这一评论。

割平面法是当之无愧的实际计算冠军，所以当然可以考虑把它用在TSP的复杂性分析中。

可惜，割平面法在最差情形下的性能如何，似乎不容易一窥究竟。1987年，我和Vašek Chvátal、Mark Hartmann共同证明，分支切割法有一种强的形式，需要至少$2^{(n/72)}/n^2$步操作，才能解决哈密顿回路问题的一道专门构造的棘手题目，而其他TSP题目甚至可能需要更多的步骤。不过该分析并未封杀所有可能性，新类型的割平面仍有可能大大改进运

行时间界限。当前，人们正在搜寻性能更佳的分支切割法实现，而搜寻新的割平面恰是不错的理论目标，两者完全可以相辅相成，同步进行。

9.4.5 近优路线

一旦证明P≠NP，对于好的TSP算法便无需再抱任何希望。幸好，旅行商尚有一线生机。例如，第4章描述了Nicos Christofides提出的基于树的启发式算法，它保证能给出成本不超过最优路线之1.5倍的路线。假如能够改进这一结果，得到1.01倍的近似算法，保证给出与最优解之差小于1%的路线，又将如何？届时，该算法的快速计算机实现会成为许多相关应用的得力工具，用途超乎想象。

即便无法用来确定P与NP问题的结论，这样的近似算法必定代表了一条值得探究的路子。然而，为研究它们，就必须限制旅行成本的取值范围，排除困难的“是/非”判定问题。[1]Christofides本人的做法也就是通用标准做法：假定旅行成本是对称的，而且满足三角不等式，即对于任意3座城市*A*、*B*、*C*，从*A*到*B*和从*B*到*C*的旅行成本之和不能小于直接从A到C的旅行成本。满足这种自然条件的题目则称为TSP的度量（metric）题目。

1. 一般的哈密顿回路问题可以成功编码，方法是在图中存在对应边的各对城市之间的旅行成本都赋值为0，不存在的各对城市之间成本赋值为1。则哈密顿回路的成本为0，非最优解的路线成本均至少为1。因此，如果存在某种方法，对于任意给定的近优路线与最优路线之间误差百分比要求，都能返回符合要求的近优解，那么这种方法就能为“是/非”问题提供答案。

Christofides算法最早出现在卡耐基梅隆大学的一篇论文里。当时是1976年，他的结果似乎相当简单。可是30年过去了，一直没有改进，可见他的结果并没那么简单。确实，要找到多项式时间的α倍近似算法，满足α小于1.5且适用于一切度量题目，仍是亟待解决的难题。

公平公正地说一句，有必要指出，其实也许根本不可能超越Christofides。支持这一消极论点的依据是，Christos Papadimitriou和Santosh Vempala业已证明，除非P=NP，否则没有多项式时间的α倍近似算法能解决TSP的度量题目且满足α小于1.0045[2]。因此，他们的研究扼杀了在P≠NP时获得极好近似算法的幻想。真正的计算壁垒在哪儿？更接近1.0045还是更接近1.5？没人知道。没法缩小这一范围固然使人不爽，不过有趣的未解研究课题列表上面可以再添一项了。

2. 准确值是220/219。

9.4.6 Arora定理

普林斯顿大学的Sanjeev Arora证明了一个重要定理，既揭示了近似方法的愿景，也揭露了它的陷阱。他证明，无论选取怎样的α值，只要超过1.0，就存在适用于欧氏距离TSP的多项式时间α倍近似算法。[1]注意欧氏距离的情形与度量情形不同，后者除非存在计算最优路线的多项式时间算法，否则根本不能指望这么好的结果。这是Arora定理的乐观一面，很有意思，表明欧氏距离的TSP可能比一般的度量题目更容易搞定。

1. Arora, S. 1998. J. ACM 45, 753–782. 另参见：Mitchell, J. 1999. SIAM J. Computing 28, 1298–12309.

下面指出近似方法的陷阱。虽然Arora定理是出色的理论结论，但当α值接近1.0时，近似算法的运行时间却会显著增长，实验结果更是很不理想。其实，近似方法一贯如此，因为要获得高质量的解，就需要非常精细地划分空间，同时还要以延长搜索时间为代价。要把Arora的几何方法巧妙转化为实际TSP工具，还有待后人出手解决。

9.5 非计算机不可吗

尼尔·斯蒂芬森（Neil Stephenson）在科幻小说《飞越修道院》（*Anathem*）中描述了一台神奇的外星机器，它能解决“偷懒的佩里格林”（Lazy Peregrin）问题——这个名字是他虚构的，本质就是旅行商问题。

> “说的就是那道题，一个漫游者需要拜访好多个数学圈，那些数学圈四处散落在地图上。”
>
> “没错，问题要求找出经过所有目的地的最短路线。”
>
> “我大概明白了，”我说，“可以列一张表，穷举出所有可能路线。”
>
> “但那样永远也算不完的，”奥若罗说，“在葛罗德圣人的机器里，你可以推广这种情况，设计某种一般化模型，配置机器，使它实际上同时检验所有可能路线。”

“葛罗德圣人的机器”有魔力，但现实世界里，同时检验所有路线的实体设备也已针对TSP构想出来，甚至存在某些测试实例。别忘了，图灵风格的计算绝对不是解决旅行商问题的唯一手段。

9.5.1 DNA计算TSP

如何制作“葛罗德圣人的机器”？1994年，美国南加州大学教授Leonard Adleman提出了一种生物思路[1]。Adleman是一位曾获大奖的计算机科学家，“RSA密码系统”中的“A”指的就是他的鼎鼎大名。他的TSP解题装置在分子层面运作，力图利用极少量DNA存放无数信息。

1. Adleman, L. M. 1994. Science 266, 1021–1024.

Adleman解决的题目是TSP的变形，属于哈密顿路径问题。输入为一个图，目标是找到一条路径，起始和终止顶点均已指定，要求该路径周游所有顶点，但不考虑旅行成本问题。他在实验中采用了一道包含7座城市的例题，如图9-4所示，从城市0出发，到城市6结束。在这个例子里，大多数边都类似于道路的单行线，哈密顿路径必须遵从箭头所示方向。只有顶点1和2之间、顶点2和3之间的两条边允许双向通行，其他边则都只允许单向通行。

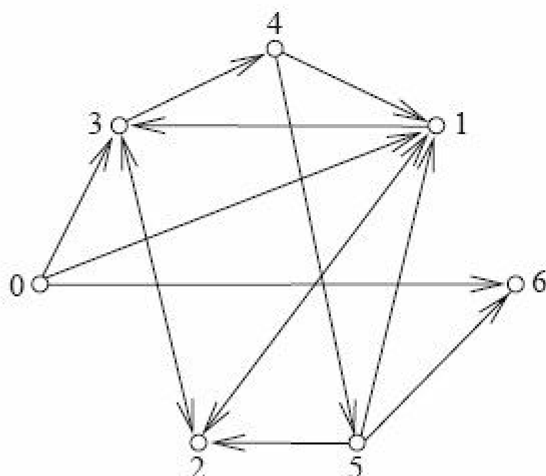


图**9-4**　有向哈密顿路径问题

针对该问题，Adleman发明了一种分子编码，赋给每个顶点一个随机的20位DNA字母串。例如，赋给顶点2的DNA字母串为

TATCGGATCG|gtatatccga

而顶点3则为

GCTATTCGAG|cttaaagcta

如上所示，20位字母串的顶点标签可以分成两组，视为相连的两个10位字母串标签，第一个标签全由大写字母组成，第二个则全是小写字母。这样一来，要表示从顶点a指向顶点b的边，就可以把a的后半标签和b的前半标签组合成一个字母串。例如，从顶点2到顶点3的边就是

gtatatccga|GCTATTCGAG

两段字母串分别取自顶点2的后一组和顶点3的前一组。这条规则只有两种例外情形，就是边包含起始顶点0和终止顶点6的时候。此时不再只用一半的10位字母串，而是使用整条20位字母串标签。如果一条边允许双向通行，则将它拆分成两条单向边，分别指向一个方向。

下面进入Adleman解法的第二步，给出一种能把各边连成一条路径的机制。除了起点0和终点6，对于其他每个顶点，写出字母串标签的互补DNA序列。在按照一致的方向连接两条边的时候，这样的序列起到“夹板”的作用。以顶点3为例，它的互补序列可以连接边(2, 3)和边(3, 4)，因为该序列的前半段与边(2, 3)的后半段互补，而序列的后半段与边(3, 4)的前半段互补。

在Adleman领导的实际实验中，研究者在实验室制备了分别与边和“夹板”对应的DNA链的多份副本，并将它们混合起来。经过七天的缜密工作，他们最终发现了一条得到完整哈密顿路径的双链。

9.5.2 细菌

Adleman的实验需要有位科学狂人在实验室里辛辛苦苦工作一个星期，而处理DNA原本就是活的生物的拿手好戏，所以改用其他生物替人干活的主意值得一试。由一群本科生组成的研究小组就实现了这个点子，在细菌体内构建合适的DNA序列，成功解决了哈密顿路径问题的一道小例题[1]。研究小组的成员来自美国大卫森学院、约翰逊 C.史密斯大学、西密苏里州立大学和北卡罗来纳中央大学四所院校。

1. Baumgardner, J. et al. 2009. J. Biol. Eng. 3, 11.

当然，使用细菌计算机自有其难处：既然对应于路径的DNA包埋在活体内部，研究者要如何发现并识别它呢？上述研究小组的答案是，利用荧光性质，迫使体内DNA确为路径的细菌菌落发光。他们用一道三座城市的题目演示了荧光方法。当两条有方向的边在哈密顿路径里相接时，红色荧光和绿色荧光叠加，得到发黄光的细菌菌落。实验结果如图9-5所示，展示了这一思想。左图中，DNA链的初始顺序构成了哈密顿路径，细菌生长之后便出现了许多黄色菌落；右图中，DNA链的初始顺序是错误的，但经过一系列突变，有一定数目的菌落“变节”了，显示出不该属于它们的黄色。

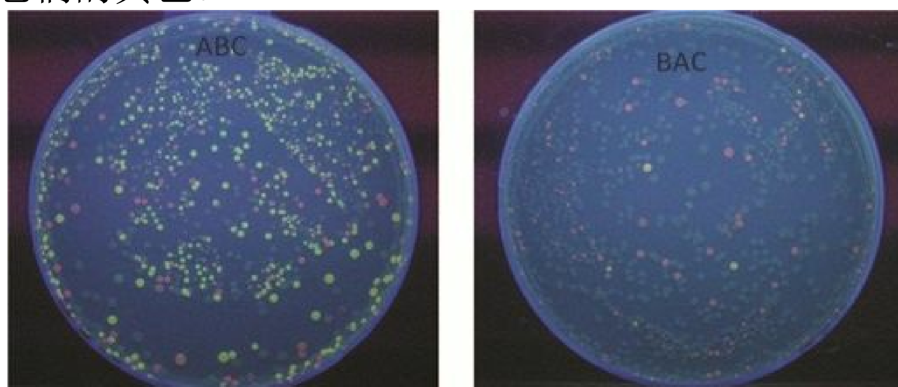


图**9-5**　哈密顿路径的细菌计算实验，**Todd Eckdahl**供图

好吧，三座城市确实太少了，几乎算不上TSP。不过毕竟总要从简单的例子讲起，而且这里的重点在于思想很巧妙。细菌培养时，数目会呈指数增加，这一特点或许可以用在规模更大的计算中，届时需要使用更加精细的方法来筛选整理得到的菌落。

9.5.3 变形虫计算

从低等的细菌走向食物链的上游，接下来登场的是单细胞的变形虫。一个来自日本的研究团队创立了使用变形虫解决TSP题目的通用做法——变形虫计算可解决一般的TSP题目，而不仅限于哈密顿路径问题。[1]该变形虫计算机的核心部件如图9-6所示，左侧是一只变形虫，右侧是一枚具有星形开口的塑料构件。将变形虫放在构件中，它会逐渐变化自身形状，完全填充构件中心的星形空间。因为变形虫会避开光源，所以可在构件的每条星形放射状分支内打开或关闭光源，借此引导变形过程。要点是变形虫有能力响应变化环境并调整至最优外形，而研究者恰要利用这一能力。

1. Aono, M., et al. 2009. New Generation Comp. 27, 129–157.

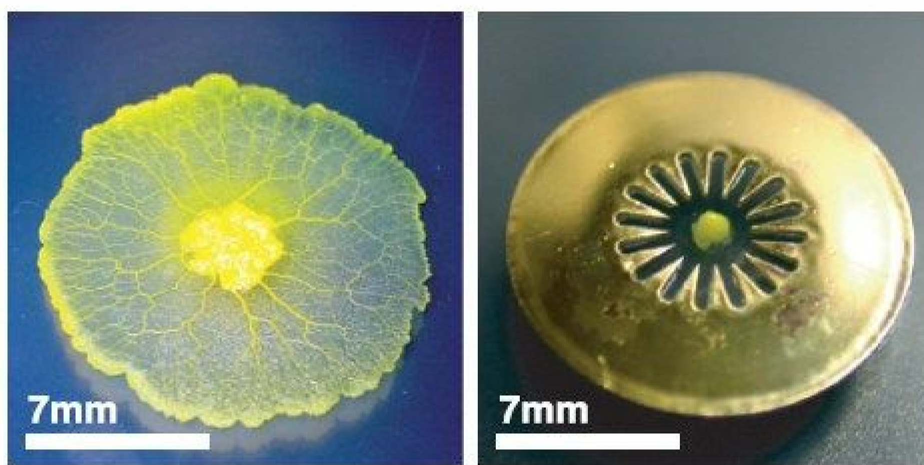


图**9-6**　单细胞变形虫和星形围栏构件，青野真士供图

四座城市TSP的解题程序在变形虫计算机上的实现如图9-7所示。这里用到的星状结构具有16个放射状分支，每座城市都由4个分支表示。城市A对应的分支标记为A1、A2、A3、A4，数字表示城市A在总路线中所处的位次，如果变形虫选择A2，则A就是路线中第二座城市。实验开始1小时14分钟之后，变形虫开始将一只伪足探入A4分支。这时，计算机程序在A1、A2、A3处打开光源，以保证满足城市A只能在路线中出现一次的规则。同理，B4、C4、D4处的光源也同时开启，以保证满足只能有一座城市处于路线中第四位的规则。不过，上述光照步骤在变形过程中不会始终严格维持，这是为了在短暂关闭光源的时候，让变形虫有机会重新选择其他城市作为路线的第四位城市。为了引导变形虫获取较短路线，选定A4后，距离A较远的城市的1、3位分支会开始闪烁照明，阻碍变形虫选择这些分支。最终，变形虫到达稳定状态，给出的解对应于路线D-C-B-A。

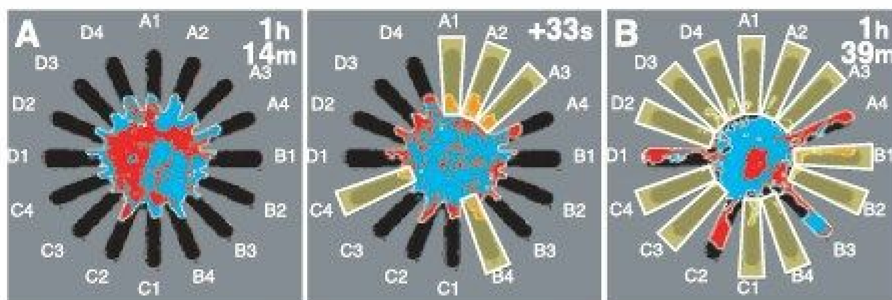


图**9-7**　变形虫求解**4**座城市的**TSP**题目（青野真士供图）

和细菌求解的例子差不多，这里只有四座城市，没有真正的计算难度可言。这项研究的最重要之处在于，研究者发明了一种由生物构成的新型可工作计算机。

9.5.4 光学

“葛罗德圣人的机器”还有一种基于光学的方案，用于求解哈密顿路径问题。2007年，这种解法在互联网上一鸣惊人。[1]它的思路是构造图对应的物理装置，光纤对应边，延时装置对应顶点。机器运行时，光到达一个顶点，经过固定的延迟时间之后分散成多束光线，分别沿着自该顶点引出的各边传播。延迟装置用来提供特征信号，用以标志光线已传过相应顶点。令D等于每个顶点恰经过一次时整条路线的延迟总时长。解决哈密顿路径问题时，我们从起始顶点发射光线，检验光线是否在D个单位时间之后到达终止顶点。

1. Haist, T.,W. Osten. 2007. Optics Express 15, 10473–10482.

表面上看，这种光学解题机器解决*n*座城市的问题时，需要的步骤数目正比于图的顶点数目。它之所以一举成名，正是因为人们看中了这一点。然而，罗马尼亚计算机科学家Mihai Oltean经过仔细分析得出结论，选择延迟时间长度的时候，若想给出可分辨的特征信号，D必须至少为2^n个单位时间。[2]目前，接收时间差如果小于10^{-12}秒，示波器就无法区分，因此解题时间至少是$2^n \times 10^{-12}$秒，按照指数速率增长。幸好，数字还是很小，中等规模的题目也能很快解决。

2. Oltean, M. 2008. Natural Comp. 7, 57–70. Oltean的研究也出现在2006年的一份会议论文集中，先于Haist和Osten的研究。

据Oltean估测，如果图具有33个顶点，那么一秒钟就够了，还不错。

不过同样据他计算，实现该例的延时装置需要的线路长达8×10^{11}米。此外，要解决一道包含几百个顶点的题目，需要的光子数目比太阳每年输出的光子数目还多。这下可不好办了。

9.5.5 量子计算机

无论是基于DNA、细菌、变形虫还是光学原理，上述TSP解题程序虽然都有完全一步到位的优点，但是也都离不开随城市数目增加而呈指数增长的资源。要制造货真价实的“葛罗德圣人的机器”，或许需要抛开生物和古典物理，另谋出路。事实上，我们发现一种更有可能成功的思路要利用量子物理学特性。第一个提出把量子物理用在计算设备中的人是Richard Feynman。

量子计算设备最为基本的组成部分是量子位（qubit）。传统计算机使用0/1二进制位表示信息，量子位与传统二进制位类似，但很特别。它能够存放值0，也能存放值1，还能同时存放0和1两个值。根据量子力学的神奇特性，如果有100个量子位，那么它们合起来就能同时编码2100种可能性，因此确实有可能实现真正的“一步到位”，一次检验所有TSP路线。

世界各地的研究小组都在孜孜不倦地奋战。他们希望能克服物理和工程方面的困难，从而开发出能正常运转的量子计算机。根据Peter Shor的知名结果，这样的计算机将给出整数因子分解问题的快速解法。不过，如果认为数目众多的量子位就能保证轻松求解旅行商问题，那你就错了。诚然，一百万个量子位足以编码周游1000座城市的每条路线，问题出在物理学上。虽然所有路线都能同时表示，但是真正考察所有量子位状态的时候，却只剩下一条路线，其他路线全都消失得无影无踪。既然有这么糟糕的现象，我们怎么可能让机器选出最优路线呢？问得好，眼下人们确实不清楚是否可能。

Scott Aaronson在《科学美国人》上发表了一篇文章，对此作了讨论，并描述了量子计算的可能局限。[1]关于NP完全问题的多项式时间量子算法，他针对其难度写出了如下评论：

1. Aaronson, S. 2008. Sci. Am., March, 62–69.

> 假如存在这样的算法，那么算法必然会以前所未见的方式利用问题的结构，而同类问题的传统高效算法也是如此，两种利用方式大致相同。仅凭量子的神奇特性本身，将无法获得成功。

量子计算机具有迷人的神奇性能，类似图灵机的传统计算机难以望其项背。但是它究竟能否派上用场，能否显著增强给旅行商指路的能力，目前依然没有定论。

9.5.6 闭合类时曲线

Aaronson的文章还讨论了几种猎奇的计算模型，其中之一是我的最爱——时间旅行解题程序。原理很简单。在稳定可靠的计算机上开始运行Concorde程序，并设置程序将很久之后找到的解传回到当前时刻。这样一来，一眨眼就能完成，但是读者自然要问：我们从这台时间机器上接收到解之后，能否直接关掉计算机？如果立刻关机，那么Concorde是怎么找到那个解的？

有些人认真琢磨这些思想，想到“闭合类时曲线”的概念，即路径穿越时间和空间之后首尾相接，构成闭合的环路。[1]如果存在这样的环，那么旅行商就会在曲线中途被追上，我们或许就能在返程途中得到解。

1. Deutsch, D. 1991. Phys. Rev. D 44, 3197–3217.

9.5.7 绳子和钉子

回到现实世界，让我们脚踏实地，别忘了Dantzig、Fulkerson和Johnson使用的那台实体装置，还有20世纪初期货真价实的旅行商背后的助手团队。他们使用的都是钉绳法，即用图钉在地图上标出目的城市，再用一条绳子勾勒出可能的路线。绷紧绳子便可以测量路线长度，所以这种装置比纯手工计算更快，又比穿越时空的量子计算机更容易搭建，直至今日仍是最实用的TSP实体辅助解题工具。

第10章 谋事在人

> 我们的做法是这样的：选出一组才华横溢的年轻人，让他们了解某些著名NP问题的历史与理论。旅行商问题就非常合适。
>
> ——Charles Sheffield，1996年[1]

1. Sheffield, C. 1996. Mood Indigo thoughts on the Deep Blue/Kasparov match. IBM.

无论是旅行商、律师、牧师、作家还是游客，多年来都一直在计划路线，更不必提所有网球运动员，在漫长的训练课程结束后，还要把一地的球都捡起来。人脑既然拥有此类经验，那么能否胜任求解一般形式TSP的任务，成为非计算机的解题平台呢？

10.1 人机对战

1997年，世界冠军Gary Kasparov与IBM公司的深蓝（Deep Blue）计算机之间展开一场国际象棋比赛。这场较量俨然如同知名体育赛事，参赛双方均吸引了热情的支持者。有人希望机器发展再受到几年的牵制，于是为人类冠军加油助威；也有人是软硬件爱好者，因而投入计算机阵营。科幻作家Charles Sheffield负责为IBM公司报道该竞赛。在本质上属于计算问题的赛事中，人类居然能与大型计算机势均力敌，不相上下，这一事实令保持中立的Sheffield颇为意外。他开始思索，如果换作其他棘手的计算难题，人类能否具有同样的竞争力，比如换成TSP。

面对一场TSP的人机大决战，如果按照Sheffield的建议，不考虑特殊学习和训练，我会赌计算机取胜。这种选择一部分是因为我的个人经验。在2007年的一次数学研讨会上，Sylvia Boyd提出一道50座城市的TSP，掀起一场挑战赛，规则是所有计算一律要求人工完成。挑战赛持续了一天，我和Dave Applegate得出了获胜路线（如图10-1所示），以微弱的优势战胜另一位TSP研究者Gérard Cornuéjols。可惜的是，这条路线不是最优路线。我们研究TSP已经有20年了，可是在计算机不到一秒钟就能算出的结果面前，依然只能望洋兴叹。

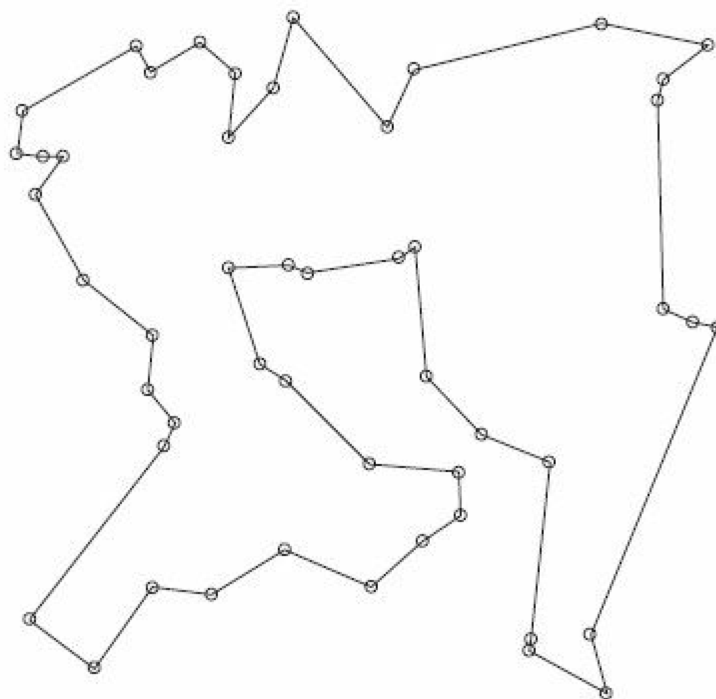


图**10-1**　数学会议上的获胜路线

10.2 寻找路线的策略

人类或许没有发现最优路线的天分，不过在其他方面还是相当优秀的。美国国家航空航天博物馆有一件展品，向参观者提出挑战，让他们试着寻找周游美国部分机场所在地的路线。展品如图10-2所示，在触控面板上，参观者可以一个接一个地选取各座机场，逐步构造出一条路线。在这样的挑战面前，对于中等大小的例子，人类参与者总是能够得到质量好的路线。尽管许多数学方法也能轻易得出质量相当的结果，但是很明显，人工求解过程中用到的直接计算步骤少得多。已有心理学研究团队对这一现象进行探究，旨在理解人类的解题能力。[1]

1. 在此项研究的帮助下，《解题学报》（*Journal of Problem Solving*）创刊，其主编为人工求解TSP领域的领军人物Zygmunt Pizlo。



图**10-2**　美国国家航空航天博物馆的**TSP**展品，**Bärbel Klaaßen**供图

10.2.1 路线之格式塔

心理学专家发现，如果对比不同质量的路线的几何形状，高质量的路线比低质量的路线“感觉对”，这或许能表明追求最小结构的人类本性。在澳大利亚阿德雷德大学，Douglas Vickers主持开展了一项实验，将这一论题表现得淋漓尽致。[1]研究中，主持人向两组参与者展示一组相同的TSP题目，规模分别为10座、25座及40座城市（每种规模有两道题目），但给出不同的实验指导：最优化组得到的指令是找出每道题目的最短路线，格式塔组则是找出“整体看起来最自然、最迷人、最优美的路线”。实验结果表明，两组得到的路线质量竟然相差无几。事实上，寻找路线水平最高的实验参与者来自格式塔组，她是一名时装设计师，在6道题中找出了5道的最短路线。

1. Vickers, D., et al. 2001. Psychol. Res. 65, 34–45.

10.2.2 儿童找到的路线

加拿大维多利亚大学的Iris van Rooij领导了一项实验，探究在面对相同题目时，儿童的寻找路线能力与成人的差别有多大。[1]借助这类实验，研究者可以对比考察知觉（感知）能力与认知能力。这是因为人们认为，低龄童在寻找路线时，将主要依赖他们对好结构的知觉能力。

1. van Rooij, I., et al. 2006. J. Prob. Solv. 1, 44–73.

实验参与者是7岁和12岁的小学生及一组大学生，小学生将得到贴纸作为奖励。TSP测试集由随机生成的题目组成，规模为5座城市、10座城市和15座城市，每种规模有5道题目，以参与者找到的路线长度超出最优解的百分数作为其表现分数。实验结果如表10-1所示，由此可见，从儿童到成人，表现有所提高，但是就连低龄童也能找出相当短的路线。

表**10-1**　儿童与成人找到路线长度的平均最优性差距

城市数目	**7**岁儿童	**12**岁儿童	成人
5	3.8%	2.5%	1.7%
10	5.8%	3.4%	1.7%
15	9.4%	5.0%	2.7%

10.2.3 凸包假说

英国兰卡斯特大学的James MacGregor和Thomas Ormerod有不同的研究重点，即点集的整体形状对寻找路线的指导程度如何。[1]为衡量点集的整体形状，可以考虑橡皮筋围住这一组城市的样子，如图10-3所示。

1. MacGregor, J. N., T. Ormerod. 1996. Percept. Pyschophys. 58, 527–539.

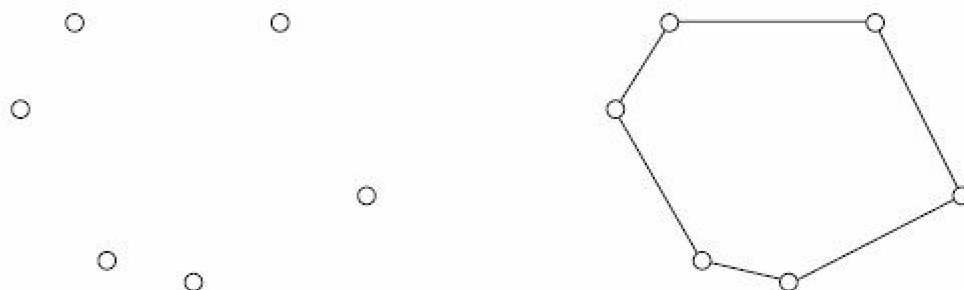


图**10-3**　一个点集的凸包

橡皮筋所在的曲线就是这组城市的凸包边界。边界本身通常并非路线，但容易验证凸包规则：最优路线需要避免自交，所以路线经过边界各点的顺序必须和凸包边界经过边界各点的顺序相同。如图10-4所示，在33座城市的宝洁公司TSP题目中，凸包边界上共有12个点，它们在边界上的顺序与在最优路线上的顺序相同，其余城市的顺序则是由边界向内侧拐入短的子路径而得。这里把不在边界上的城市称为内部点。

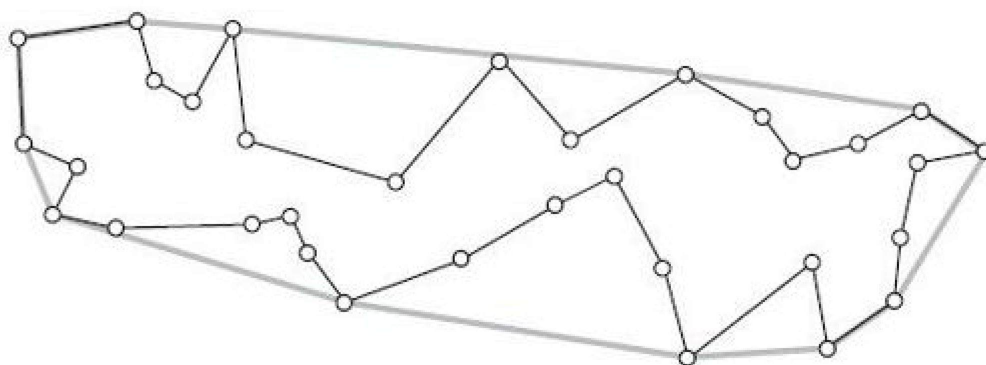


图**10-4**　沿凸包顺序的最优路线

仔细分析10座和20座城市TSP题目的人工解题实验结果之后，MacGregor和Ormerod得出结论：寻找TSP题目近似解的难度是由内部点的个数决定的。另外，他们还写道："这里给出的证据表明，人类实验者求解的依据是感知TSP题目的全局空间特性，尤其是凸包边界的空间特性。"这种假说的真实度几何？人工解题领域争议不休。多数专业讨论的焦点都落在实验研究使用的数据集类型，但是同样存在一般性的疑问，即人类利用的解题策略究竟是从整体到局部还是从局部到整体。如

果是从整体到局部，人类会首先感知整体结构，然后做出局部选择，将各座城市填入整体结构的合适位置；反之，如果是从局部到整体，则人类会首先做出局部分析（比如聚类分组），然后才尽力将局部信息以最好的形式组合成整体路线。

10.2.4 实地TSP题目

在上面这些人类参与的实验中，TSP题目都是视觉呈现的，即出题形式或为在纸上描绘路线，或为在计算机屏幕上找出路线。Jan Wiener和德国图宾根大学的一个研究小组反其道而行之，针对实地TSP研究人类表现。他们的实验要求参与者在一间长8.4米、宽6.0米的房间里访问一系列目的地。[1]实验中，25个长方形柱体摆在地上，每个柱体上有一枚彩色符号。参与者得到的指示包括一个出发点和若干符号（最多为9种），他们需要到达所有符号所在柱体并最后返回出发点。结果再次支持了人类擅长求解小规模TSP的结论，该实验的报酬是每小时8欧元（约合64元人民币）。

1. Wiener, J. M., N. N. Ehbauer, H. A. Mallot. 2009. Psychol. Res. 77, 644–658.



图**10-5**　人力求解**TSP**的实验，**Jan Wiener**供图

10.3 神经科学中的TSP

对于规模非常小或者大部分城市都落在凸包边界上的TSP题目，实验参与人总是可以正常解决，彼此之间几无差别。然而，城市数目达到20座以后，参与人表现的个体差异立刻就变得非常明显，Vickers等人的研究中的那位时装设计师就是一例。在Vickers领导的另一项实验中，题目规模继续扩大到50座城市，各人找到的路线质量便出现了一致性的差异。[1]研究者还指出，TSP能力和标准非言语智力测验分数之间存在一定相关性。

1. Vickers, D., et al. 2004. Pers. Indiv. Differ. 36, 1059–1071.

连线

在类似TSP的问题上，人类实验者的表现差异一直是神经心理学临床研究的依据。Halstead-Reitan成套测验（或称HRB神经心理成套测验）中的连线测验就是一个精彩的实例。[1]

1. Reitan, R. M., D. Wolfson. 1993. *The Halstead-Reitan Neuropsychological Test Battery: Theory and Clinical Interpretation*. Neuropsychology Press, Tucson, Arizona, USA.

连线测验的第一个环节如图10-6左侧所示，有25座带标号的城市。测验要求被试者画一条路线顺次连接各座城市，完成速度应当尽可能快，主试会随时指出错误。正确路线如图10-6右侧所示，显然不是周游所有城市的最优路线，不过也算是没有自交的较短路线。测验的第二个环节与之大同小异，只是把城市由标号改为1，A，2，B，…，12，L，13的混合标记。这项分为两部分的测验由美国军方心理学家于20世纪40年代发明，现在通用的计分系统由Reitan设计，评分标准完全基于被试者完成任务花费的时间。

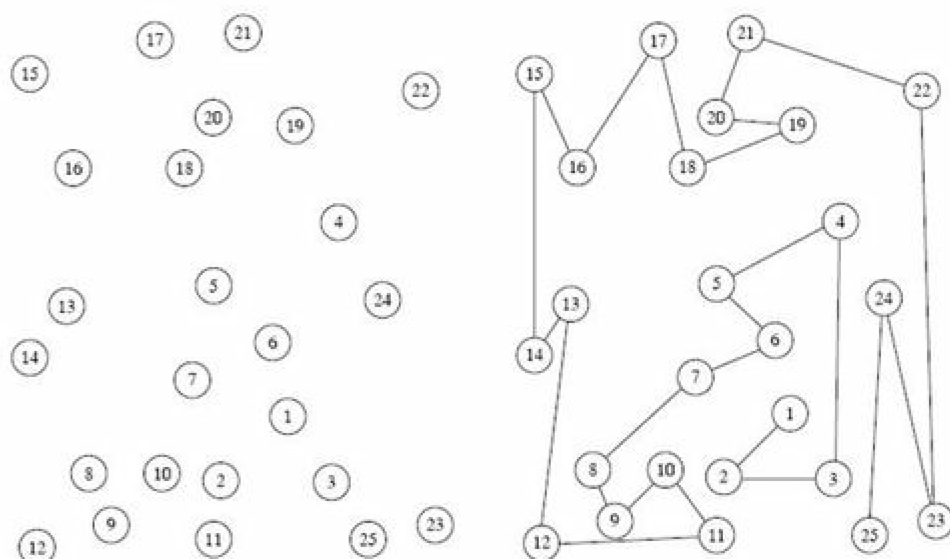


图**10-6**　连线测验（第一部分）

无数实验结果表明，连线测验可以鉴别出遭受大脑损伤的患者，而且灵敏度很高。事实上，1990年的一篇综述称，它是国际神经心理学会各成员机构中最广为使用的测验。[2]耐人寻味的是，连线测验的两个环节总是使用特定的城市位置分布，似乎没有靠谱的方法能生成具有良好临床性质的其他分布。

2. Butler, M., et al. 1991. Prof. Psychol.-Res. Pr. 22, 510–512.

10.4 动物解题高手

人类是优秀的TSP解题者，那么动物的水平如何呢？Emil Menzel在1973年的研究里提出了这一问题，他的研究对象是一群黑猩猩。[1]Menzel设计了一个精巧的方案，诱导被试黑猩猩走出一条高效的路线——毕竟没法再用8欧元或者贴纸充当奖励了。试验开始时，6只黑猩猩关在场地边缘的一个笼子里。驯兽员将选定的黑猩猩带出笼子，领着它在场地上到处走动，与此同时，一名助手在随机地点藏下18份水果。然后，黑猩猩回到笼子里。经过两分钟的等待后，六只黑猩猩同时被放出。之前选定的黑猩猩会利用它对食物藏放位置的记忆，在同伴盲目搜寻找到好吃的水果之前，尽快将它们收为己有。

1. Menzel, E. W. 1973. Science 182, 943–945.

该研究中一只黑猩猩Bido采取的路线如图10-7所示。Bido在场地边界标为“起点”处出发，最后到达标为“终点”处，图中几处箭头标识了实际的前进方向。它没拿到4份水果，但是在整体上，它借助对食物位置的记忆找出了相当好的路线。

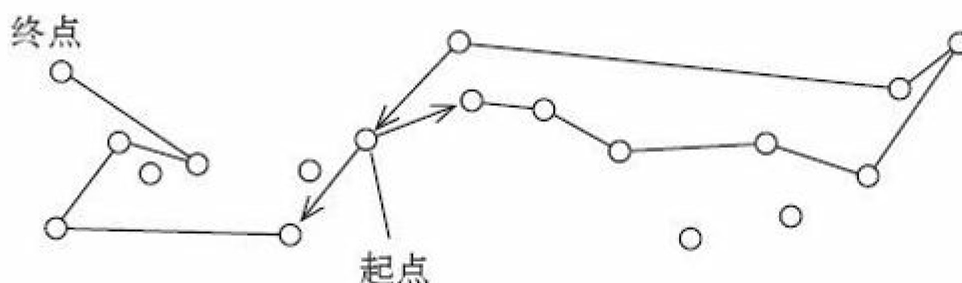


图**10-7**　黑猩猩**Bido**找到的路线

动物求解TSP的其他研究用到的实验动物包括绿猴、狨猴和大鼠。已发表的实验均用到了实地TSP题目，要求动物从分散的地点收集食物。新罕布什尔大学的Brett Gibson则独辟蹊径，主持了一项鸽子实验。[2]实验鸽子经过训练，学会通过啄触控面板上显示的位置来确定一条哈密顿回路，如图10-8所示。等到显示出的所有城市都访问过后，鸽子就会得到两粒鸟食。结果，鸽子选择的路线比随机路线短得多，但一般比不上最近邻路线。第二次研究中，为了鼓励鸽子找到更短的路线，规则改为只在路线质量足够高时才能得到鸟食。鸽子不想失去零食，于是提高了解题水平，找出了测试集中小规模TSP题目的好解法。

2. Gibson, B. M., et al. 2007. J. Exp. Psychol. Anim. B. 33, 244–261.

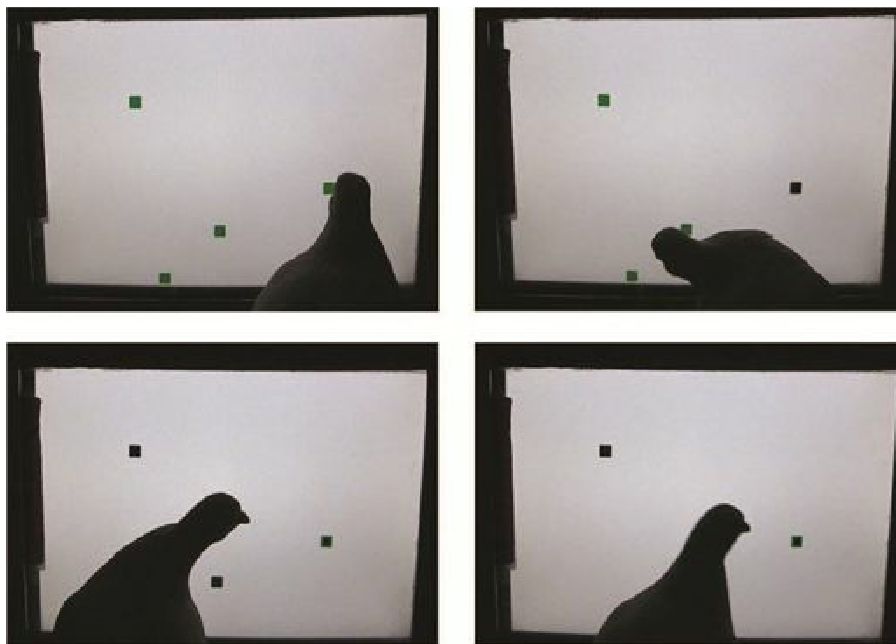


图**10-8**　鸽子求解**TSP**，**Brett Gibson**供图

在上述研究中，由于问题规则简洁，研究人员能够设计不寻常的实验，而TSP解题任务也提供了测试不同动物实验对象的空间认知能力的好方法。

第11章 错综之美

> 数学一直是复杂图案（在圈外人眼里则是神秘图案）的源泉。
>
> ——Jaroslav Nešetřil，1993年[1]

1. Nešetřil (1993).

数学家如果形容某项研究极为美妙，完全不表示这种美可以实体化。TSP也不例外。也许有条周游一组点的路线形状很优美，但是数学家并不会因此着迷。TSP的几何结构和复杂度结合之美，才真正令数学家流连忘返。

尽管如此，一些迷人的美术作品中也出现了TSP的影子。有些画作甚至成功抓住了旅行商引人入胜的数学本质。

11.1 Julian Lethbridge

见到Julian Lethbridge创作的旅行商画作时，我很高兴。Lethbridge是一位知名的当代艺术家。他的作品收藏于美国华盛顿国家美术馆、纽约大都会博物馆、芝加哥艺术学院和伦敦塔特美术馆。阅读以下评论，可以体会到他的创作风格。

> Julian Lethbridge似乎决意表现出相交曲线所能表现的一切。
>
> ——《纽约时报》，1995年[1]

1. *New York Times*, November 17, 1995, page C60.

> Lethbridge的抽象艺术是理性的，常常以数学原理和自然法则为基础。

——ULAE，1997年[2]

2. *Proof Positive: Forty Years of Contemporary American Printmaking at ULAE, 1957–1997*. Corcoran Gallery of Art, Washington, D.C., 1997.

> 他的艺术语言往往起源于随机或天然形成的图案，诸如碎玻璃和蜘蛛网之类。

——Paula Cooper Gallery，1999年[3]

3. Paula Cooper Gallery, Julian Lethbridge, March 26–April 24, 1999.

> 直线、曲线、格线、环线、规则的线、有序的线、无序的线、凹线、凸线、粗线、细线、黑白的线、彩色的线……五花八门的线条是Lethbridge美术体系的基础组分，也是构建他笔下自然界的基本元素。

——Art in America，2007年[4]

4. Harris, S. 2007. Art in America 95, issue 10, page 214.

透过这些评论可以读出数学家熟悉的思想。事实上，我曾在纽约与Lethbridge见过一面，就他的美术作品展开讨论，但话题很快转向了数学。他有意了解数学家的审美趣味，也抱有同样的直觉。

他在谈起旅行商系列作品时，向我解释说，他在一本期刊里看到了TSP的描述，立刻被好路线的简洁思想和简约空间深深吸引。图11-1和图11-2是该系列中的两幅优美作品。在《纽约时报》和《巴尔的摩太阳报》（*Baltimore Sun*）的文章中，评论家把这些作品同Jasper Johns的地

图主题油画和Jean Dubuffet的作品《虚幻的群体》（*Compagnie Fallacieuse*）相比。[5]TSP爱好者恐怕会有些失望，因为评论家比较的作品都没有像Lethbridge的画作那样安排画面空间。

5. *New York Times*, November 17, 1995, page C60; *Baltimore Sun*, March 31, 1996.

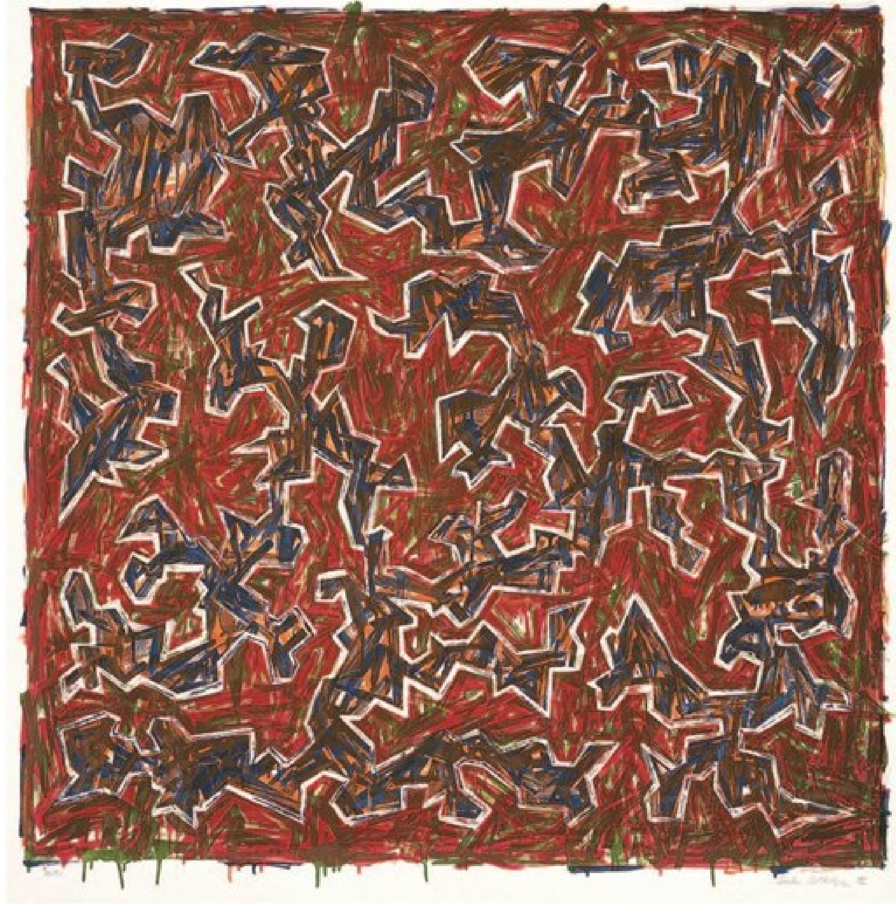


图**11-1**　《旅行商》，**Julian Lethbridge**，**1995**年，版画，**1.11**米×**1.07**米（**43.75**英寸×**42**英寸）（**Julian Lethbridge**和**United Limited Art Editions**供图）



图**11-2**　《旅行商（其四）》，**Julian Lethbridge**，**1995**年，油画（亚麻布），**1.83**米×**1.83**米（**72**英寸×**72**英寸）（**Robert and Jane Meyerhoff Collection**收藏，**Adam Reich**拍摄）

在图11-1中，请注意路线没有自交，形成了简单闭曲线，又称为若尔当曲线（Jordan curve），将空间划分为两部分，一部分有界，另一部分无界。在我们看来，有界的区域是曲线内部，无界区域则是曲线外部。这幅画把路线涂成白色，用纹理强调路线的内外两侧，表现出这种空间划分。

图11-2描绘的是同一条路线，表现形式却大不相同。这次，Lethbridge同样没有画出城市位置，而是以各城市为中心向四周挥笔涂抹色彩，以此表现出五花八门的周游路线。这幅大型油画现藏于美国国家美术馆，属于Meyerhoff Collection。

11.2 若尔当曲线

球面上的路线没有内外之分。球面会分为两部分，像相邻的拼图一样镶嵌在一起。图11-3左图为Robert Bosch设计的雕刻工艺品《拥抱》（*Embrace*），上面刻出了一整条若尔当曲线，对称性显而易见。它的表面并不是球面，而且由于有最外侧的浅色环形，实际上可以分辨出内侧区域和外侧区域。尽管如此，图案的中心对称性依然令人称奇。2010年，这件工艺品在数学艺术展（Mathematical Art Exhibition）中荣获一等奖，颁奖机构为美国数学学会和美国数学协会。



图**11-3**　左图：**TSP**雕刻工艺品《拥抱》；右图：多重对称环，**Robert Bosch**供图

Robert Bosch的名字在第1章已经出现过，那道含有100 000座城市的《蒙娜丽莎》TSP题目正是出自他之手。Bosch在美国欧柏林学院数学系工作，现任系主任一职，是一位数学家。数学最优化问题当年是他博士学位论文的研究方向，如今又成为他从事美术创作的工具。[1]

1. Bosch, R. 2010. Embrace. *2010 Joint Mathematics Meetings Art Exhibition Catalog*.

> 我对一件作品有了创意之后，先把创意转化成一道数学最优化问题，然后解题，得到答案，看看自己是不是满意。如果我满意了，就到此为止。否则，就修改原来的数学最优化问题，重新解题，得到答案，看看这次结果如何。我经常要反复好多次才能得到自己满意的作品。这些都是因为我喜欢数学最优化——我喜欢它的理论，喜欢它的算法，也喜欢数不清的应用题。

Bosch很喜欢几种最优化方法，包括求解TSP的精确算法和启发式算法。他借助最优化方法创造了无数TSP艺术品，比如第1章的《蒙娜丽

莎》以及下一节会出现的波提切利名画《维纳斯的诞生》。Craig Kaplan是加拿大滑铁卢大学的计算机科学家。他们两人合作开发了精细复杂的方法，可以巧妙设定城市位置，从而能用一条好的TSP路线有趣地重现一幅画作。[2]

2. Kaplan, C., R. Bosch. 2005. Proceedings of the Bridges 2005 Conference.

雕刻工艺品《拥抱》的图案环环相扣。利用Bosch-Kaplan方法，可以创作出相应且合理的TSP画作，但是在得到的若尔当曲线中，相邻的环臂可能位于空间的同一区域，即同处于内侧区域或外侧区域。这时就轮到Bosch出场修改题目了。他用到一种整数规划方法，被他本人称作“随心所欲弯折曲线”（bending the curve to our will）。[3]给定TSP路线构成的若尔当曲线，他选取自己希望放在曲线两侧的两个点。这一几何要求等价于叙述“连结两点的线段必须与TSP路线相交奇数次”，而后一条件可以添加进TSP的整数规划模型中，作为附加的约束条件。因此Bosch向模型中加入这条异侧约束，使用整数规划解题程序，解决得到的新最优化问题，然后再看看生成的若尔当曲线是否称心如意。

3. Bosch. R. 2009. Proceedings of the Bridges 2009 Conference.

《拥抱》的底材是四分之一英寸（约0.6厘米）厚的金属板，内侧区域由不锈钢构成，外侧区域则是黄铜。一台水切割机[4]用来沿着726座城市的TSP路线切割这两种金属板，得到内外两侧区域的两套金属图案构件，分别配对，就制作成两个版本的《拥抱》。在切割过程中，每块构件上都切除掉一小条金属，在雕刻成品中构成空白区域，让原始路线一目了然。

4. 一种利用高压水流切割的机器。——译者注

Bosch还制作了一条规模更大的多重对称环状若尔当曲线，如图11-3右侧所示。他在一封电子邮件里如此介绍这份作品：

> 我向TSP题目里加入异侧约束条件，限制路线各边在圆心和边缘处要有五重旋转对称性，在中间区域则要有十重旋转对称性。我又加入了其他异侧约束条件，以得到内外侧分别染色后的错综交织之美。

这次的TSP题目包含2840座城市，因此很难准确求解相应的整数规划问题。图中用到的路线其实是用软件内置的启发式算法求得的。

Philip Galanter的TSP壁画

Bosch和Lethbridge都巧妙地使用了TSP和相应的若尔当曲线。Bosch

作品的巧妙之处在于设计中的对称性，而Lethbridge在画作纹理布局上独具匠心。相比之下，Philip Galanter则采取了鲜明直接的做法，绘制了一系列大型TSP壁画。其中一例见图11-4，他利用一条TSP路线和两种对比强烈的颜色给一面墙上色。



图**11-4　TSP**壁画，**Philip Galanter**供图

Galanter是美国得克萨斯农工大学视觉系的助理教授。他的研究创作主要围绕美术及音乐中的复杂性和自动化展开。在秘鲁首都利马举行的一次展览上，他介绍自己的作品时，对这一组TSP作品作出了如下评述。[1]

1. Galanter, P. 2009. Artist text. Artware 5 Exhibition, Lima, Peru, May 2009.

> 旅行商系列壁画另辟视角，探寻艺术的出现。每幅壁画都是针对那面墙专门设计而成的，首先生成大量随机点，然后计算机程序计算恰好经过各点一次的最短路线。结果，得到的所有壁画都有共同的视觉风格，这让人感觉有些不可思议。在自然界中，树或其他植物的美

> 则源自于大自然求解一道最优化问题：植物怎样才能利用最少的有机物资源，接收最多的阳光？这两个最优化问题的例子表明，美不是无意义的，也不是由个别天才独占的。美应该既隽永，又能让大众理解。

Galanter计划延续TSP系列设计，再创作几幅新作品，例如完成一件直径16英尺（约4.9米）的金属雕塑，再比如在一大片校园草坪上用足球场划线设备画出一条路线。

11.3 连续曲线一笔画

艺术家J. Eric Morales常住美国波特兰市，圈内人都称他为Mo。他的手绘若尔当曲线用法自成一体，这种艺术形式被他命名为“迷宫投影”（Labyrinthine Projection）。Mo只用一条连续曲线，就能在画中创造丰富的表现。他的一笔画技巧广受瞩目，相关作品出现在耐克鞋和苹果iPod保护套上，画像还得到了篮球明星迈克尔·乔丹的授权，见图11-5。

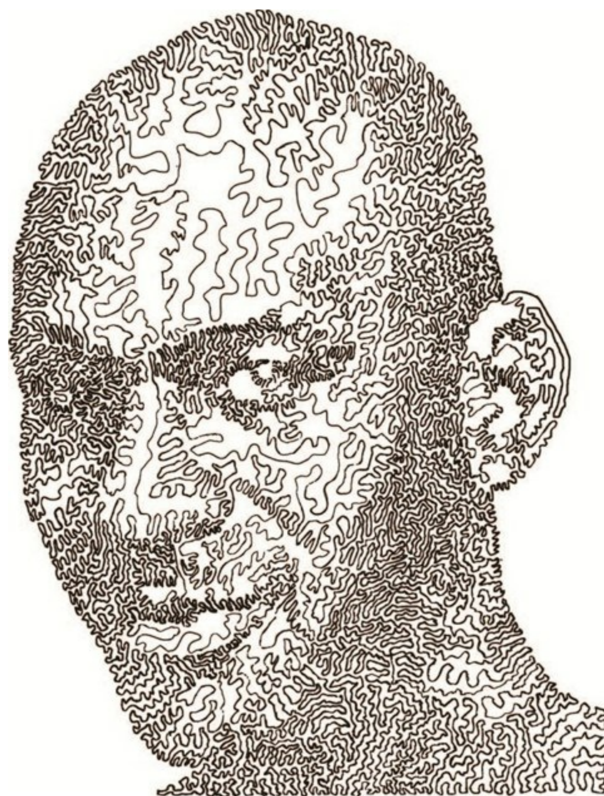


图**11-5**　面有疑色的迈克尔·乔丹，**J. Eric Morales (www.labyrinthineprojection.com)**版权所有

据Mo回忆，他小时候常玩磁性画板（Etch-A-Sketch），用一根不自交的曲线在画板上绕来绕去，画满整个屏幕，一画就是好几个钟头。后来，他学习艺术，知道了线条疏密和调子明暗之间的原则与关系。这时，他又想起了童年的一笔画，便意识到自己可以用连续曲线绘制出类似照片画质的画作，只需要操控各部分线条的间距——间距近就代表调子暗，间距远则代表调子亮。由此得到的曲线明显类似于短的TSP路线。

对于如何将TSP风格的画作拓展到其他介质上，Mo有不少点子。在

他最新的作品中，阳光穿过仿照迷宫雕塑而成的物体，投影出清晰可辨的人脸。“迷宫人”（Labyrinth Man）是他设计的另一项应用，这是一具虚拟的无头人体模型，皮肤透明。他在人体上附以一层颜色渐变的迷宫图案，从视觉上模拟其环境，并在表层传递文字信息，使虚拟人体栩栩如生。

Bosch-Kaplan的艺术作品

前一节提到Bosch-Kaplan的TSP作品，这一节又介绍了Morales的画作。Morales本人在一封电子邮件中评论了两者之间的关联。

> 我很熟悉TSP，因为我认识Craig Kaplan很多年了。他看到过一幅70英尺×40英尺（约合21米×12米）的迷宫投影图，画上是职业滑板运动员Paul Rodriguez，那是我为洛杉矶极限运动（X-Games）给耐克设计的。Kaplan的算法对我也很有意义，因为它是和我的手绘过程最接近的计算机解法。

TSP艺术项目（TSP Art project）旨在用连续曲线一笔画出原版画作的摹本。作画时需要精心选取一组城市位置，使用启发式算法，生成周游点集的短路线。以图11-6为例，这幅图对应的TSP题目包含140 000座城市。



图**11-6**　波提切利名画《维纳斯的诞生》的**TSP**版，**Robert Bosch**供图

该项目诞生于欧柏林学院，Bosch及其学生Adrianne Herman在该校发明了一种算法，按照数码图像各部分的灰度，成比例地设定城市分布。准确说来，他们将一幅图片转化为网格，根据每个格子的平均灰度，放置0到k座城市，格子几乎为白色时对应0座城市，黑色则对应k座。城市在格子内的位置是随机放置的。k值和格子的大小决定了TSP题目中城市的数目。

由一条周游Bosch-Herman点集的短路线，可得到一幅图，并能辨认出原始图像。但是密集的点往往导致路线呈锯齿状，无法充分重现原始图像的连续色调。为此，Bosch和Kaplan采用一种多遍算法，通过计算重心，重新分布城市位置，大大改进了图像的色调。计算时，参照原始图像的色调，确定城市间几何距离的权重，使得各点向色调较暗的区域集中。整体上，这一过程避免城市密度突然改变，从而可以由暗色调细腻地过渡到明亮色调。[1]

1. 该过程基于A. Secord的布局算法，用到加权沃罗诺伊图（Voronoi diagram）。

11.4 艺术与数学

本章的篇首引语摘自Jaroslav Nešetřil的一篇文章，他在文中探索了艺术家和数学家思想过程的相互关联。Jaroslav Nešetřil来自布拉格查理大学，不仅是离散数学界的重要研究者，也是成功的艺术家。他与声望卓群的专业画家Jiří Načeradský长期合作，作品之一见图11-7，展现了一条在三维空间中蜿蜒盘曲的连续曲线。



图**11-7　Jiří Načeradský**和**Jaroslav Nešetřil**，**1997**年，混合介质，**100cm×120cm**，**Jaroslav Nešetřil**供图

在那篇论艺术与数学思维相通之处的文章中，Nešetřil指出，在数学领域，图灵机的执行可以描述邱奇-图灵论题的大一统思想，即所有算法任务的形式都是统一的。因此他猜测，或许类似的“创造论题”（Creative Thesis）也对全体人类活动都成立。“所有足够深刻的活动，所有足够深刻的认知，都有显著的相似之处。相似之处体现在活动（知识）的组织方式上，体现在它们为人所知的方式上，也体现在它们彼此影响的方式上。”[1]Nešetřil总结得出上述论题的依据是过去两百年间艺术与数学的同步进展，在此期间，这两个领域都从长久的桎梏中解放出来，涌现出了新的基本形式，比如艺术界的超现实主义以及数学界的现代集合论。

1. Nešetřil(1993).

构成主义艺术与超大规模集成电路

从很久以前开始，Nešetřil便成为德国波恩大学离散数学研究所的常客，那里的负责人Bernhard Korte同样是数学界和艺术界的双栖人才。波恩离散数学研究所不仅是离散最优化的顶尖研究中心，还设有计算博物馆（Arithmeum），藏品主题为计算、美术和音乐。研究所照片见图11-8，该建筑布局巧妙，数学研究人员的工作区同时也是博物馆藏品的陈列区。



图**11-8**　德国波恩大学离散数学研究所及计算博物馆，**Bernhard Korte**供图

计算博物馆有众多展品，而它们尤其容易吸引数学家。事实上，构成主义艺术是博物馆的收藏主题之一，Bernhard Korte对此解释道："首先必须承认，我们对几何艺术和构成主义艺术形式一见倾心。为何？也许是因为数学家有纯真的心灵，所以纯粹的几何形式与构成色彩的原色一结合，便能沁人心脾。"[1]该馆的展品作者包括Josef Albers、Max Bill、Jean Gorin、Richard Paul Lohse、Leon Polk Smith和Charmion von Wiegand，等等。如果能去波恩，在美妙展品的簇拥下学习和研究，实在是不胜愉悦的体验。波恩大学在离散数学的学术领域非常活跃，显然也得益于独一无二的研究环境。

1. Korte, B. 1991. *Mathematics, Reality, and Aesthetics – A Picture Set on VLSI-Chip-Design*. Springer Verlag, Berlin.

至于应用领域，波恩的专长则是集成电路的最优化设计，即构成现代电子设备核心的计算机芯片的设计。这一研究方向称为超大规模集成电路（VLSI），复杂的VLSI芯片上约有10亿个晶体管，而波恩大学的研究人员最擅长应用离散数学来提高芯片速度，改善组织结构。此过程中的工程任务复杂得可怕，不过如果有人对背后的数学计算量一无所知，他就可能从最终产物中获得美的享受。计算博物馆的展品目录可以证明这一点，因为目录中列出了许多有趣的图片，都来自复杂的VLSI项目。这里摘录其中两例，见图11-9。几何形状代表特定计算机芯片中各组件的布局，复杂颜色则是由博物馆负责人Ina Prinz博士选取的。

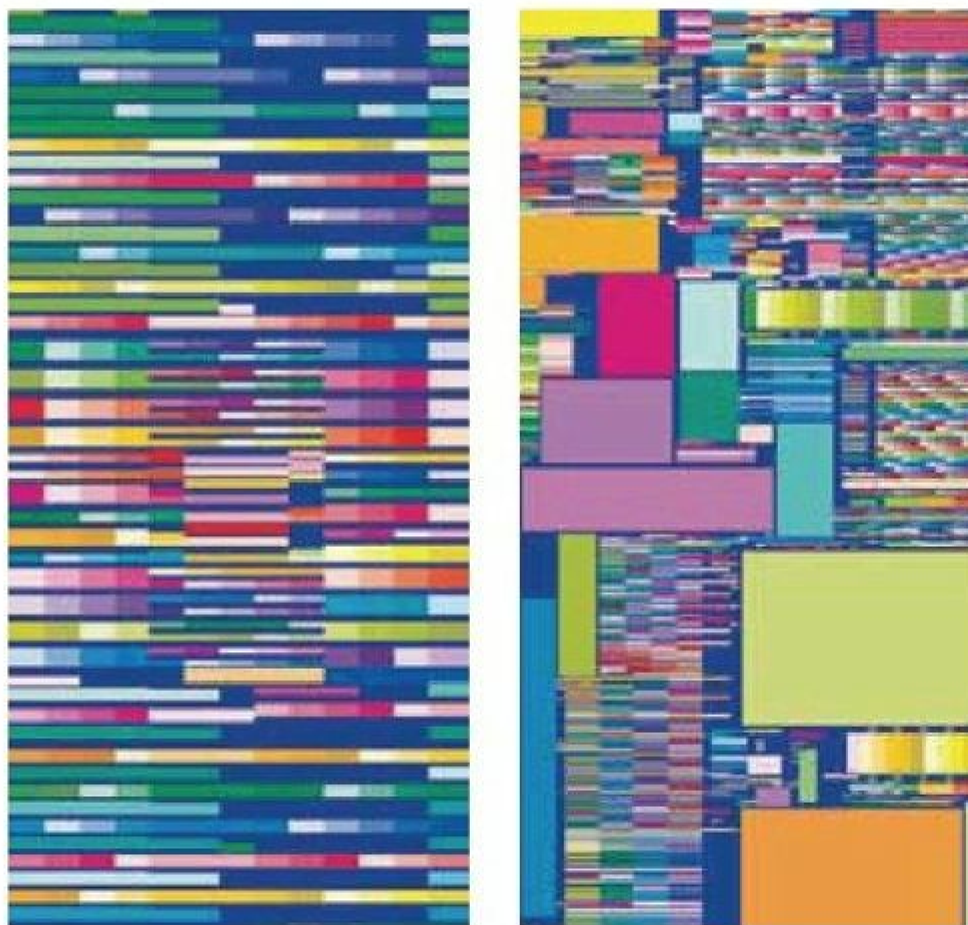


图**11-9　“Philipp”**超大规模集成电路芯片设计，**Ina Prinz**绘制

Nešetřil针对VLSI设计过程发表了下述见解，从而将此应用研究也纳入“创造论题”的范畴之内。[2]

2. Nešetřil (1993).

> 芯片设计是人类最密集的活动之一。该活动横跨若干学科，所用方法涉及计算机科学、数学、物理学乃至哲学。因此，这些活动相互

> 作用，呈现出类似艺术创作的特征，亦不足为奇。

我们对旅行商问题也有类似的体会。随着人们不断深入了解TSP内在的复杂性，TSP与艺术的密切联系也将越来越多地显露在世人面前。

第12章 超越极限

> 该问题当然还没有定论。我希望人们能在两方面完成更多的研究工作，一方面要寻找更好的计算方法，另一方面要加深数学上对它的理解。
>
> ——Delbert Ray Fulkerson，1956年[1]

1. 摘自D. R. Fulkerson写于1956年6月25日的信，收信人是法国讷伊（Neuilly）的B. Zimmern。

从今往后，TSP之美仍将继续吸引数学家和计算机科学家，这一点毫无疑问。

> Christos Papadimitriou告诉我，旅行商问题不是一道题，而是一种瘾品。
>
> ——Jon Bentley，1991年[2]

2. *New York Times*, 1991-03-12, G. Kolata.

> 它让你上瘾。不管取得了多大的进展，你总会觉得有些尚未落实的预感有可能带来重大突破，这种不安的感觉永远挥之不去。
>
> ——Vašek Chvátal，1998年[3]

3 Science Blog, 1998-06-08. http://www.scienceblog.com.

我们不教你如何戒除TSP瘾。相反，假如有机会设计糖纸，我也会毫不犹豫地在背面印上小规模的TSP难题。当然，糖不属于本书的讨论范围。

本书涉及的许多研究题目依然悬而未决，比如《蒙娜丽莎》难题和世界旅行商问题，4/3猜想，突破Held-Karp运行时间下界，还有改进Christofide近似算法界限。和别人讨论这些题目是件乐事，但你也许需要花很长时间才能取得突破，我无意隐瞒这一事实。只有非常渴望深入探究TSP计算之谜的人，才有可能对它大彻大悟。

旅行商的意义

计算机科学界大牛Avi Widgerson曾表示，复杂性理论和人类知识限界之间或许存在关联。确实，如果能证明P=NP，那么人类将大步跨进新时代，拥有能够建模并理解世界的高效计算工具。另一方面，如果事实符合多数专家的猜测，有P≠NP，那么无数重大问题将永远得不到解答，因为只要解法的运行时间呈指数增长，则用于计算的机器速度再怎么提高，都无法望其项背。

那么，我们今后要如何对付难题呢？或许在TSP计算研究的决不妥协的态度里，可以找到答案。若P≠NP，则无论是科学界还是非科学界，通用解法都将受到限制。不过，具体限制究竟为何？又如何局限人类对知识的探索？在此背景下，旅行商具有举足轻重的意义。集中力量攻克一道或许无法解决的题目，是否能够带来意料之外的结果，看看旅行商就知道。

本文到此为止，希望有读者受到鼓舞，愿意投身TSP的研究事业。这里远有百万美金的复杂性悬赏，近有实际的逐步解题算法。旅行商问题确实很难解决，但是，让我们像Rashers Ronald说的那样，不屈不挠，前进到底吧。

参考文献

[1] Albers, D. J., C. Reid. 1986. An interview with George B. Dantzig: The father of linear programming. The College Mathematics Journal 17, 293–314.

[2] Applegate, D. L., R. E. Bixby, V. Chvátal, W. Cook. 2006. The Traveling Salesman Problem: A Computational Study. Princeton University Press, Princeton, New Jersey.

[3] Arora, S., B. Barak. 2009. Computational Complexity: A Modern Approach. Cambridge University Press, New York.

[4] Biggs, N. L., E. K. Lloyd, R. J. Wilson. 1976. Graph Theory 1736–1936. Clarendon Press, Oxford, UK.

[5] Chvátal, V. 1973. Edmonds polytopes and a hierarchy of combinatorial problems. Discrete Mathematics 4, 305–37.

[6] Clay Mathematics Institute. 2000. Millennium problems. http://www.claymath.org/millennium/.

[7] Dantzig, G., R. Fulkerson, S. Johnson. 1954. Solution of a large-scale travelingsalesman problem. Operations Research 2, 393–410.

[8] Dantzig, G. B. 1963. Linear Programming and Extensions. Princeton University Press, Princeton, New Jersey.

[9] Dantzig, G. B. 1991. Linear programming: the story about how it began. J. K. Lenstra et al., eds. History of Mathematical Programming—A Collection of Personal Reminiscences. North-Holland. 19–31.

[10] Edmonds, J. 1991. A glimpse of heaven. J. K. Lenstra et al., eds. History of Mathematical Programming—A Collection of Personal Reminiscences. North- Holland. 32–54.

[11] Flood, M. 1954. Operations research and logistics. Proceedings of the First Ordnance Conference on Operations Research. Office of Ordnance Research, Durham, North Carolina. 3–32.

[12] Garey, M. R., D. S. Johnson. 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, San Francisco, California.

[13] Gomory, R. E. 1966. The traveling salesman problem. Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems. IBM, White Plains, New York. 93–121.

[14] Grötschel, M., O. Holland. 1991. Solution of large-scale symmetric travelling salesman problems. Mathematical Programming 51, 141–202.

[15] Held, M., R. M. Karp. 1962. A dynamic programming approach to sequencing problems. Journal of the Society of Industrial and Applied Mathematics 10, 196–210.

[16] Hoffman, A. J., P.Wolfe. 1985. History. In: Lawler et al. (1985), 1–15.

[17] Karp, R. M. 1972. Reducibility among combinatorial problems. In: R. E. Miller, J. W. Thatcher, eds. Complexity of Computer Computations. IBM Research Symposia Series. Plenum Press, New York. 85–103.

[18] Karp, R. M. 1986. Combinatorics, complexity, and randomness. Communications of the ACM 29, 98–109.

[19] Lawler, E. L., J. K. Lenstra, A. H. G. Rinnooy Kan, D. B. Shmoys, eds. 1985. The Traveling Salesman Problem. JohnWiley & Sons, Chichester, UK.

[20] Lin, S., B. W. Kernighan. 1973. An effective heuristic algorithm for the traveling-salesman problem. Operations Research 21, 498–516.

[21] Mahalanobis, P. C. 1940. A sample survey of the acreage under jute in Bengal. Sankhya, The Indian Journal of Statistics 4, 511–30.

[22] Menger, K. 1931. Bericht über ein mathematisches Kolloquium. Monats-hefte für Mathematik und Physik 38, 17–38.

[23] Nešetˇril, J. 1993. Mathematics and art. In: From the Logical Point of View 2,2. Philosophical Institute of the Czech Academy of Sciences, Prague.

[24] Reid, C. 1996. Julia: A Life in Mathematics. The Mathematical Association of America,Washington, D.C.

[25] Schrijver, A. 2003. Combinatorial Optimization: Polyhedra and Efficiency. Springer, Berlin, Germany.

[26] Spears, T. B. 1994. 100 Years on the Road: The Traveling Salesman in American Culture. Yale University Press, New Haven, Connecticut.

Table of Contents